MPSSDEGB
कौशल से रोजगार, समृद्धि अपार

FITT
FOUNDATION FOR INNOVATION AND TECHNOLOGY TRANSFER
भारतीय प्रौद्योगिकी संस्थान दिल्ली
Indian Institute of Technology Delhi

Computer Vision

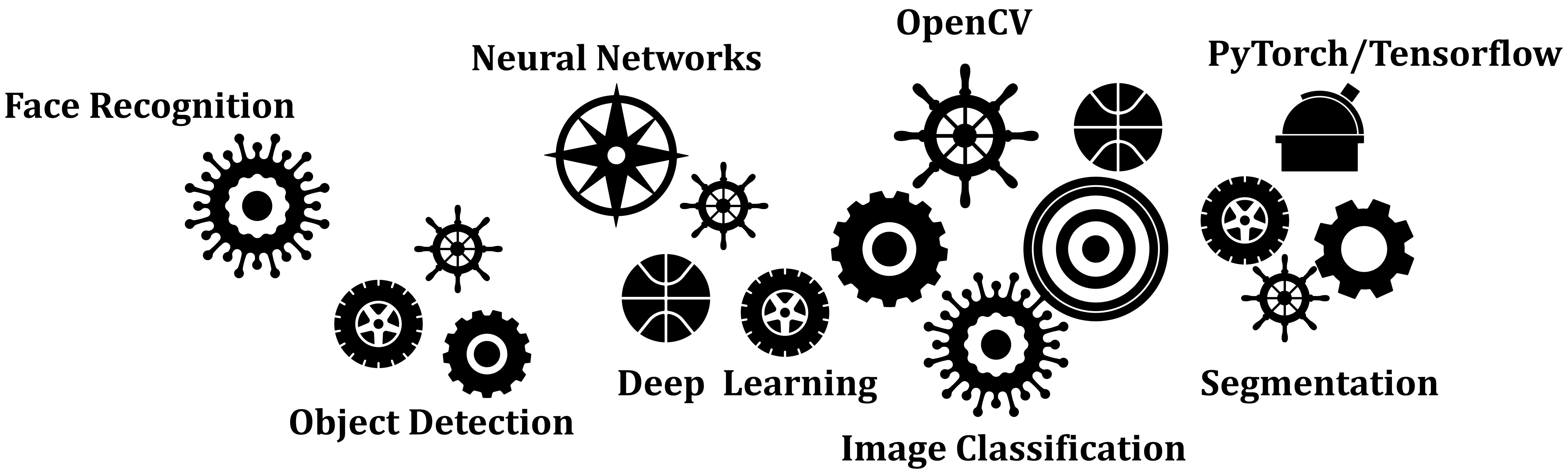


Image Segmentation

Image segmentation is the process of partitioning a digital image into multiple segments by grouping together pixel regions with some predefined characteristics. **Each of the pixels in a region is similar with respect to some property, such as color, intensity, location, or texture**. It is one of the most important image processing tools because it helps us to extract the objects from the image for further analysis. Moreover, the accuracy of the segmentation step determines success and failure in further image analysis.

You probably wonder what is the use of partitioning an image into several parts. Let's better understand image segmentation using the following example.

https://datahacker.rs/007-opencv-projects-image-segmentation-with-watershed-algorithm/

OSTU Thresholding

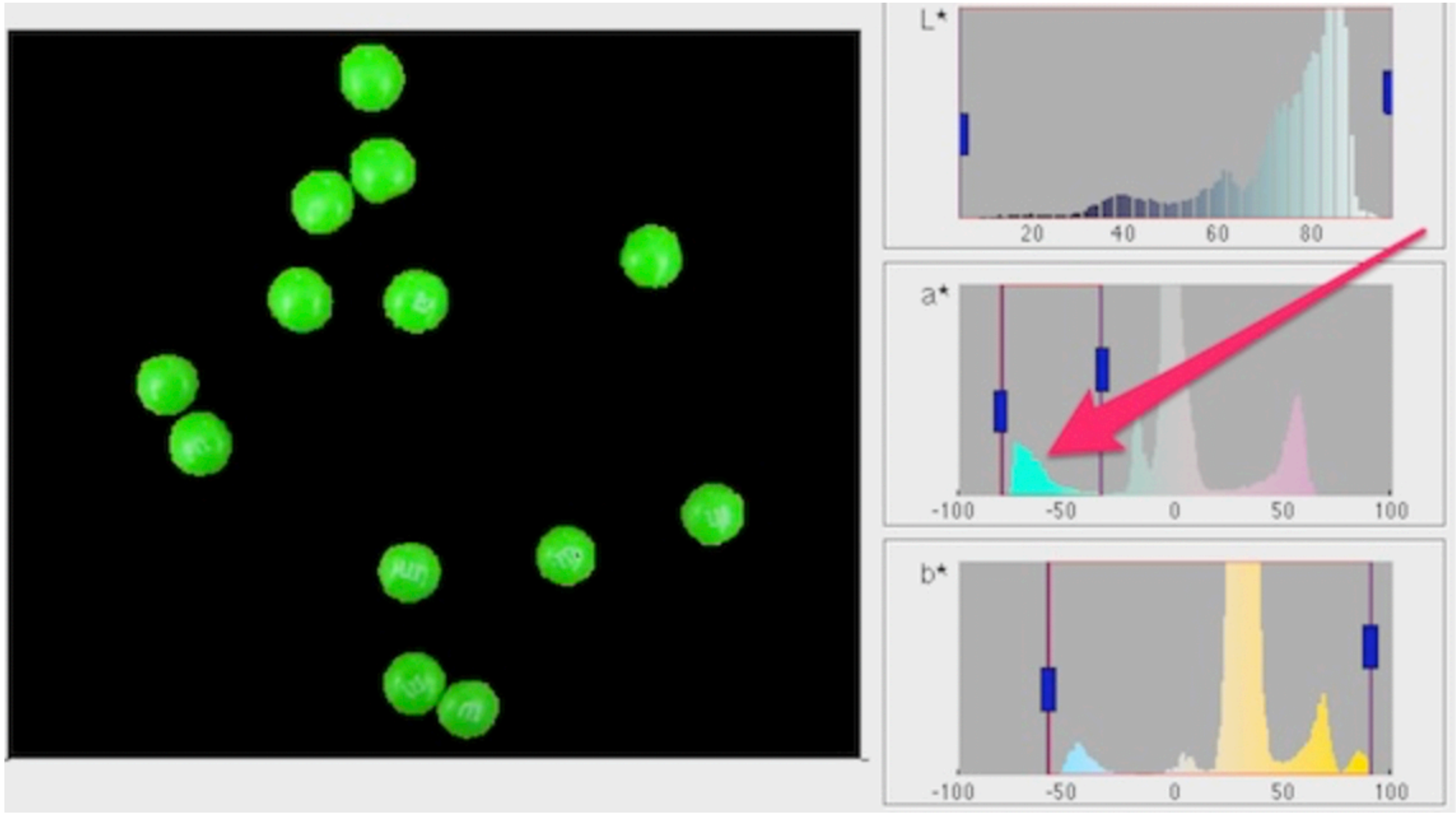
Functionality:

Threshold Determination: Otsu's Method is used for automatic threshold selection in image segmentation.

Steps:

- Histogram Computation: Compute the histogram of the input grayscale image to represent pixel intensities.
- Probability Distribution: Normalize the histogram to obtain a probability distribution.
- Cumulative Distribution: Calculate the cumulative distribution function (CDF) from the probability distribution.
- Mean Intensity: Compute the mean intensity of the image.
- Between-Class Variance: Iterate through all possible thresholds, calculating the between-class variance for each.
- Maximize Variance: Select the threshold that maximizes the between-class variance. This threshold effectively separates the image into foreground and background.
- Thresholding: Apply the selected threshold to segment the image.

Computer Vision: Pixel & Threshold



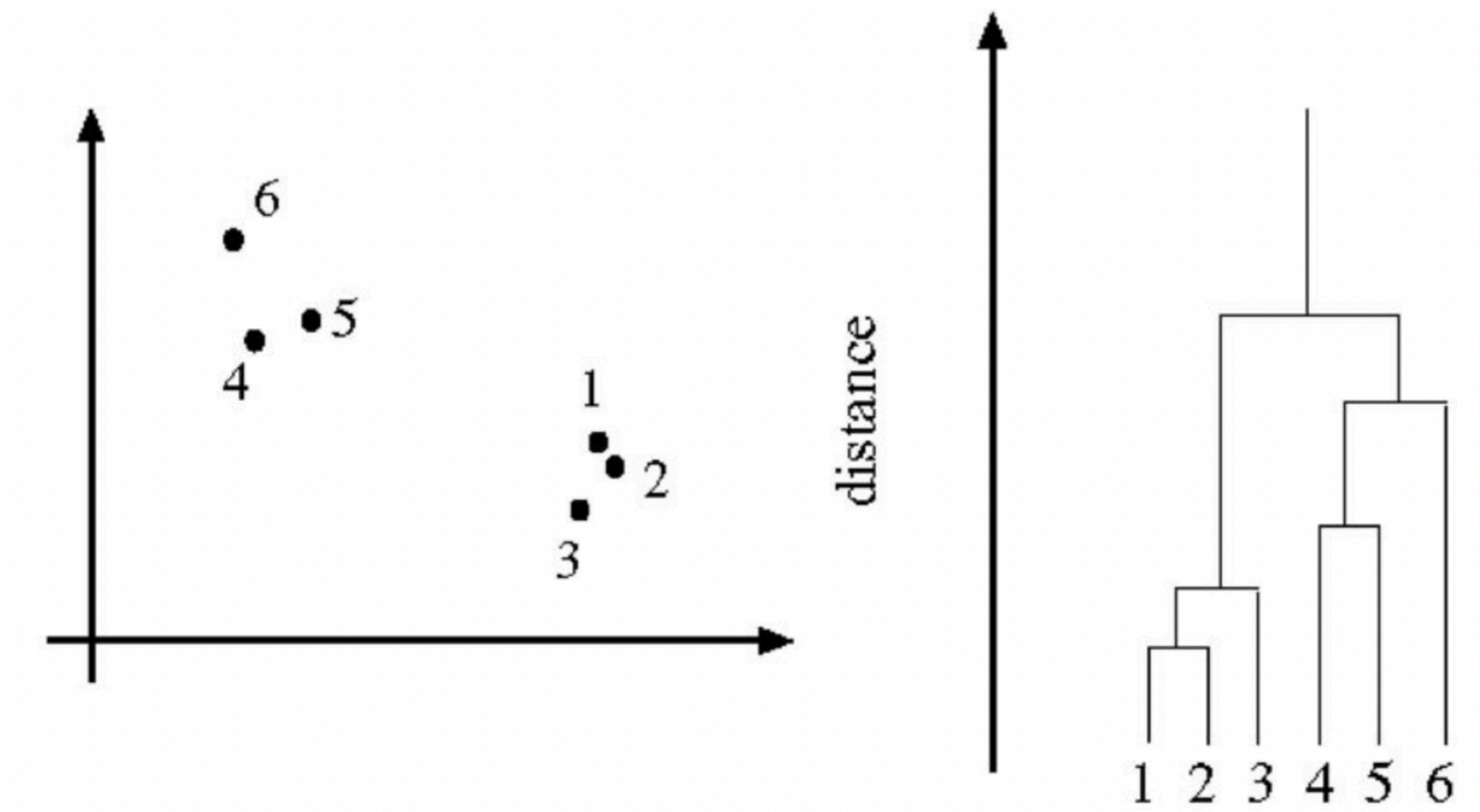
https://blogs.mathworks.com/steve/2014/04/25/color-thresholder-app-in-r2014a/

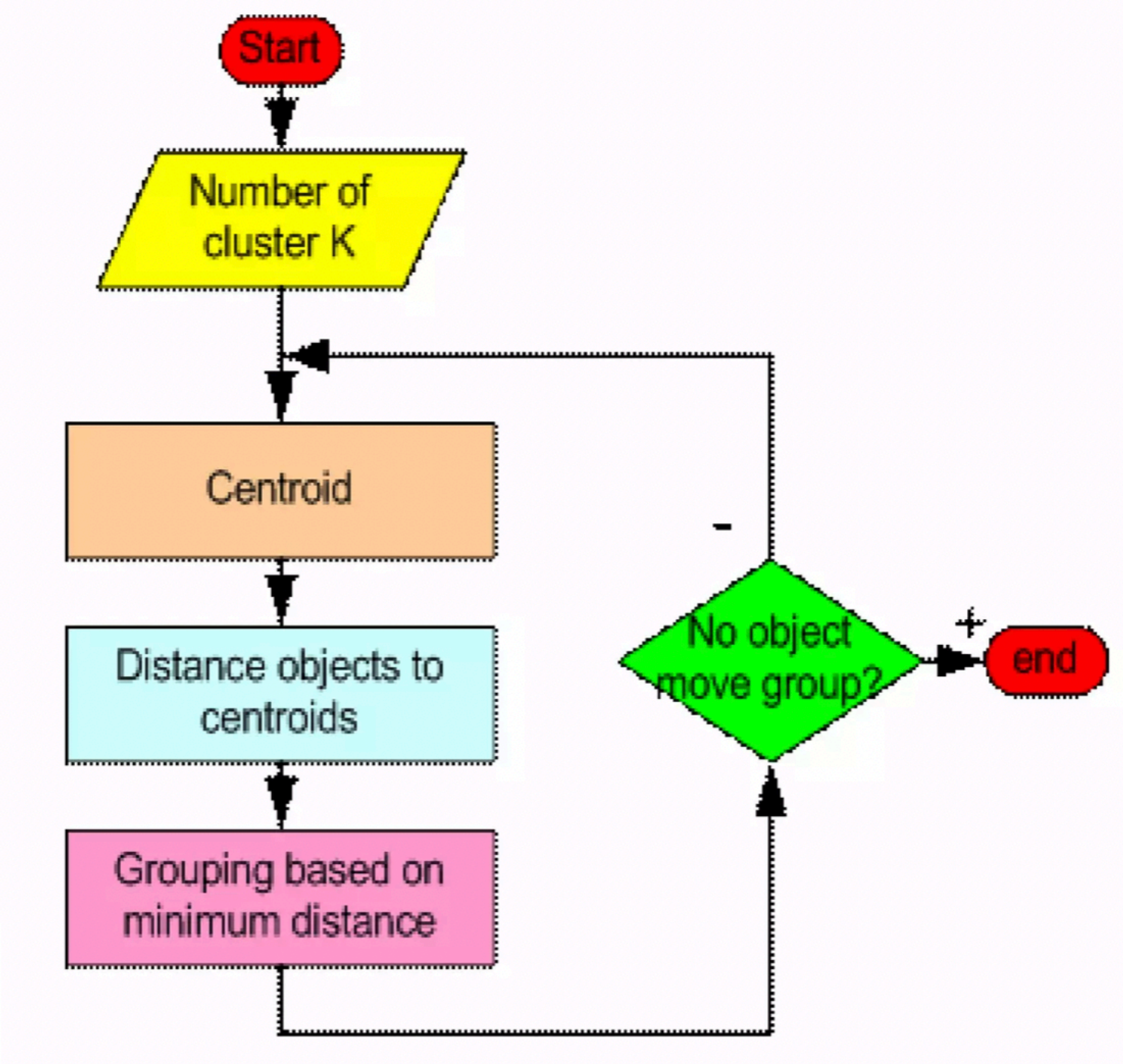
K-means Segmentation

- Choose a fixed number of clusters
- Choose cluster centers and point-cluster allocations to minimize error
- can't do this by search, because there are too many possible allocations.

- Algorithm
  - fix cluster centers; allocate points to closest cluster
  - fix allocation; compute best cluster centers
- x could be any set of features for which we can compute a distance (careful about scaling)

$$\sum_{i \in \text{clusters}} \left\{ \sum_{j \in \text{elements of i'th cluster}} \left\| x_j - \mu_i \right\|^2 \right\}$$





Parameters for k-means?

number of clusters
number of cases
centroid for cluster j
case i

$$\text{objective function} \leftarrow J = \sum_{j=1}^{k}\sum_{i=1}^{n} \underbrace{\left\| x_i^{(j)} - c_j \right\|^2}_{\text{Distance function}}$$

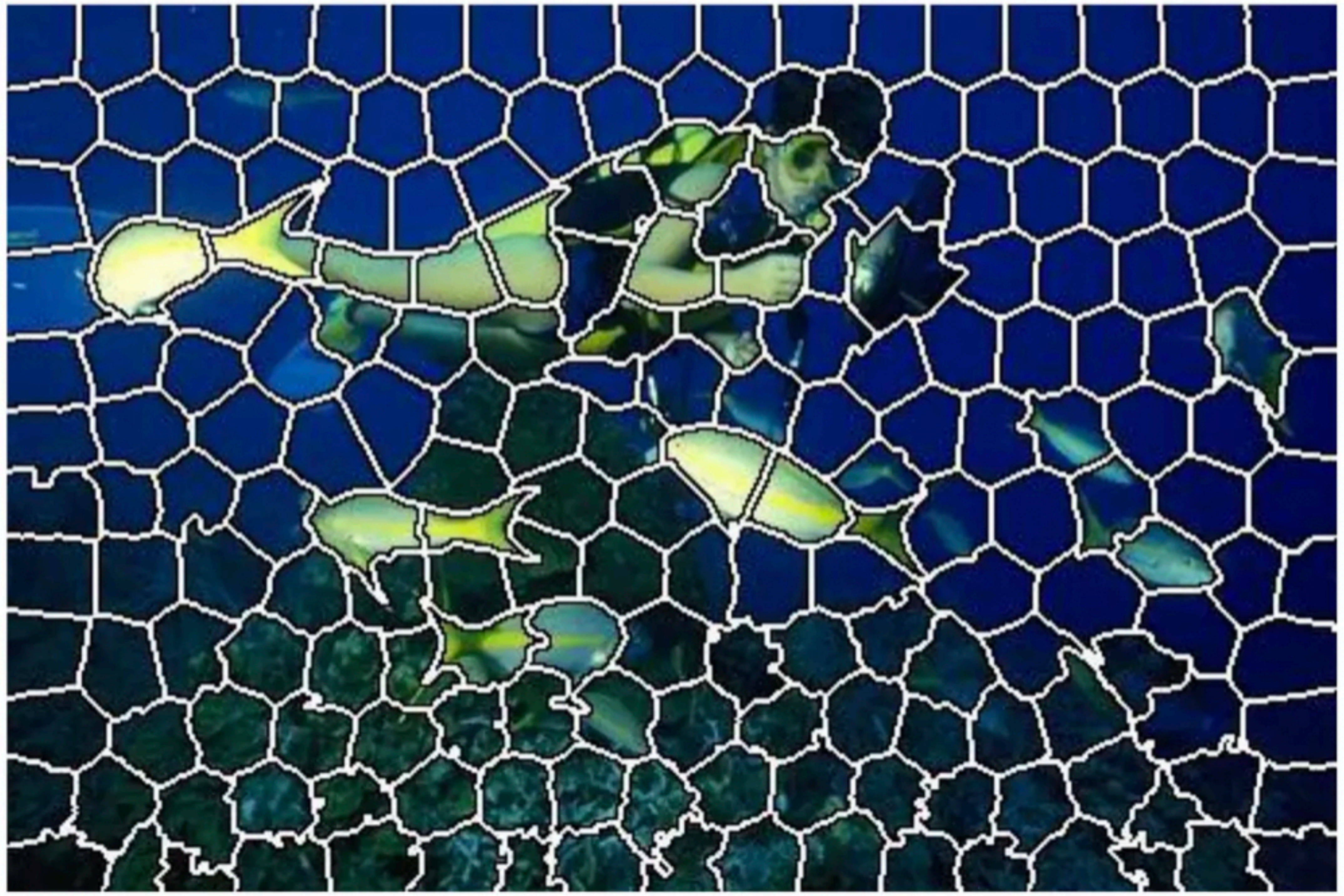
Intuitively then, the optimal choice of K will strike a balance between maximum compression of the data using a single cluster, and maximum accuracy by assigning each data point to its own cluster.

Sample images **Ground Truth** **K-means**

https://www.kdnuggets.com/2019/08/introduction-image-segmentation-k-means-clustering.html

SLIC Segmentation

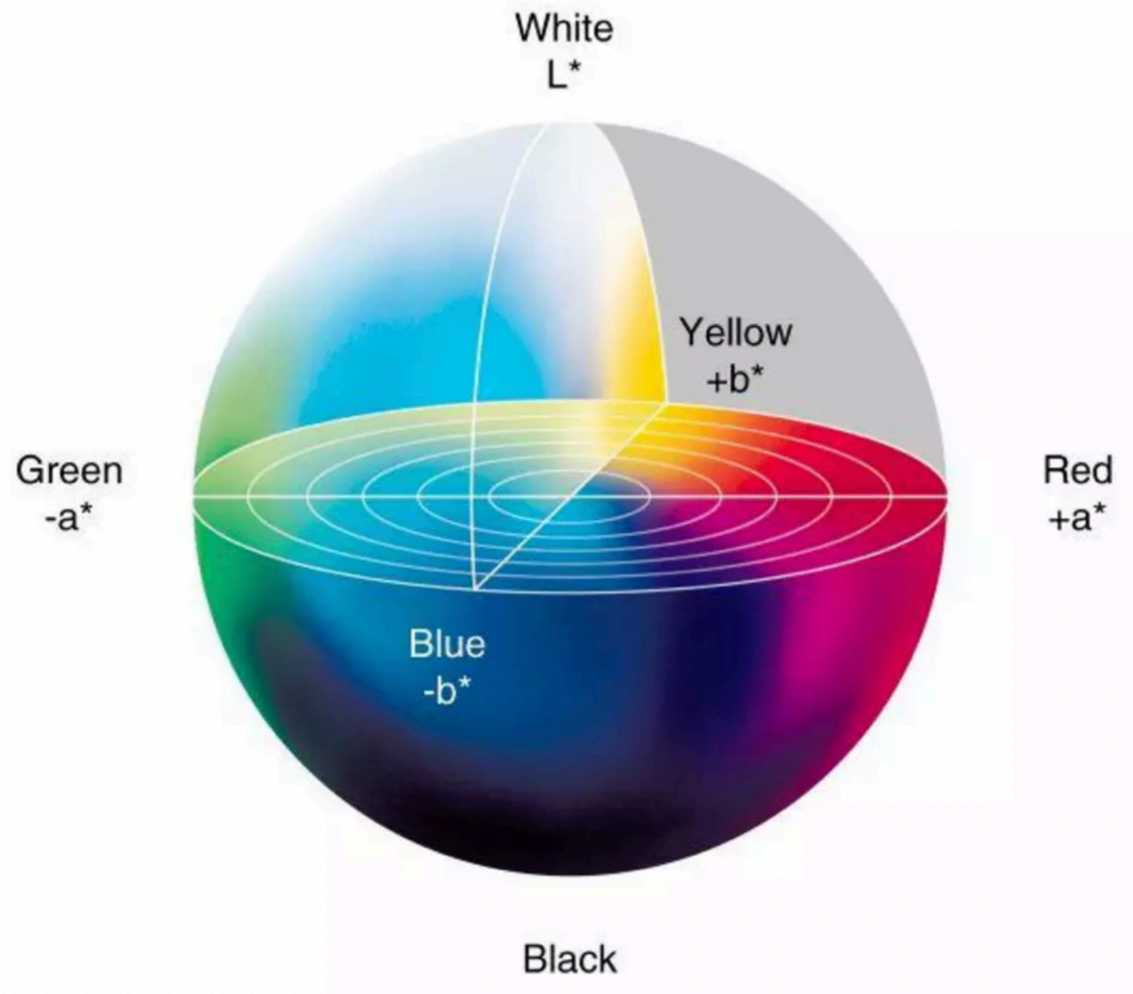
A superpixel can be defined as a group of pixels which have similar characteristics.



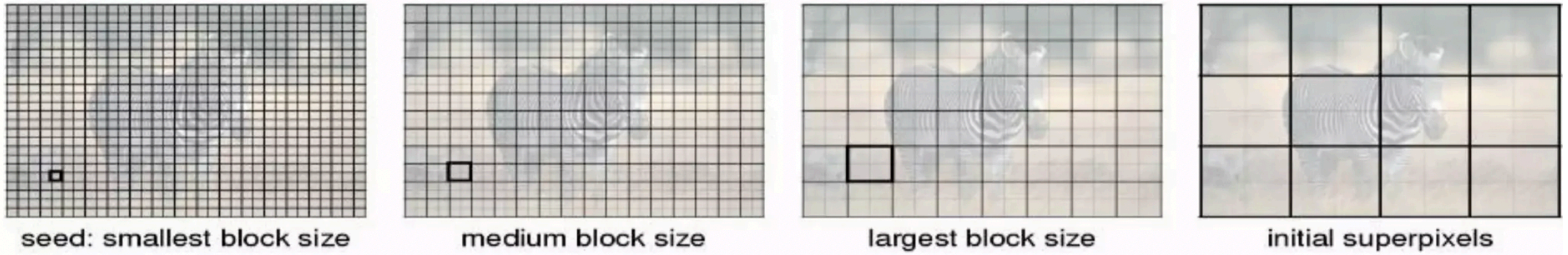
https://www.slideshare.net/ShimaFoolad1/superpixel-algorithms-whatershed-meanshift-slic-bslic-foolad

SLIC Segmentation

For color images, SLIC algorithm works in the LAB color space.



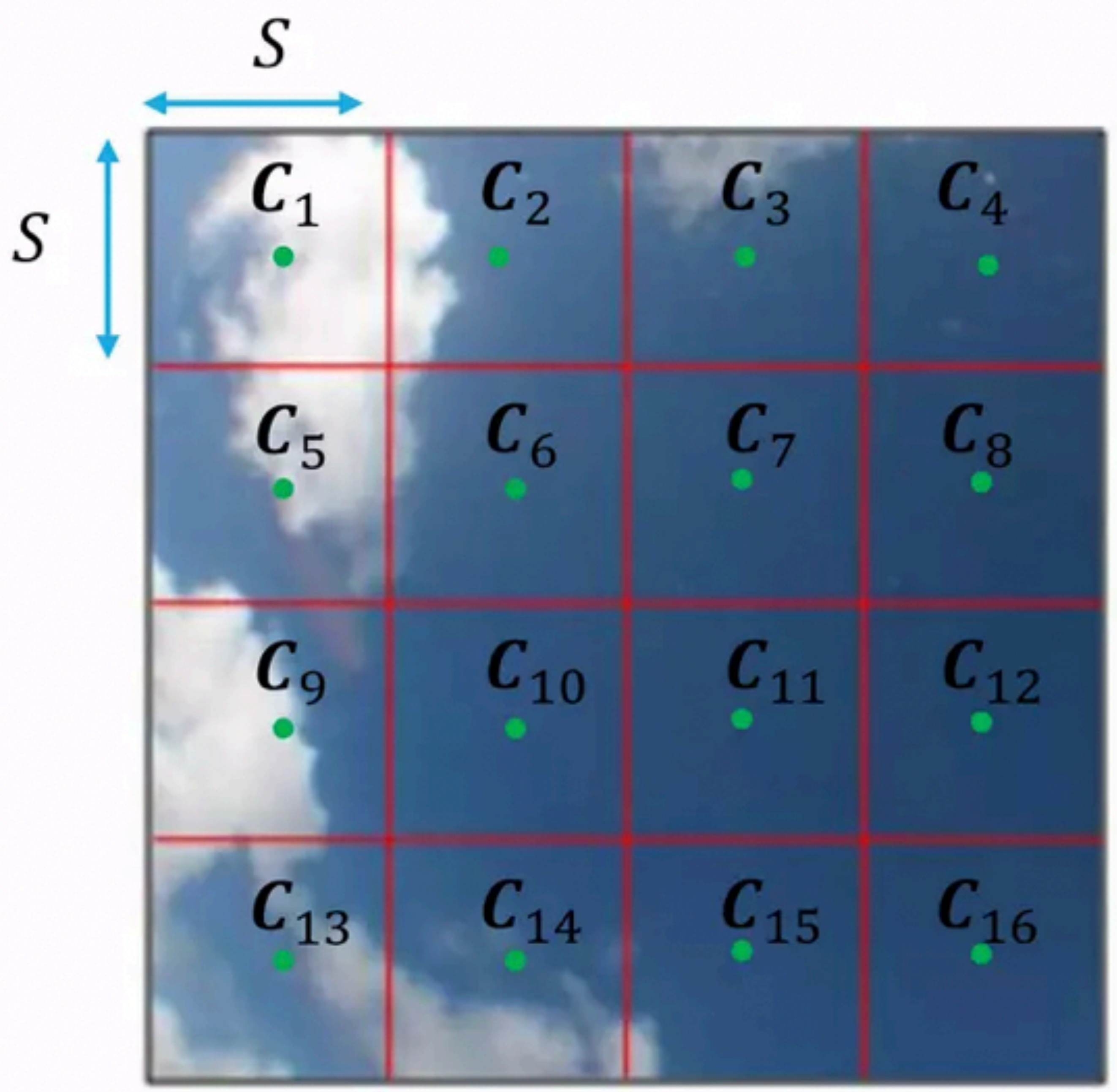
1. Initialize K cluster centers $\boldsymbol{C_k} = [\boldsymbol{l_k}, \boldsymbol{a_k}, \boldsymbol{b_k}, \boldsymbol{x_k}, \boldsymbol{y_k}]^{\boldsymbol{T}}$ on a regular grid spaced $S = \sqrt{\frac{N}{K}}$ pixels apart.
   - each superpixel has approximately $\frac{N}{K}$ pixels



https://www.slideshare.net/ShimaFoolad1/superpixel-algorithms-whatershed-meanshift-slic-bslic-foolad

SLIC Segmentation

1. Initialize K cluster centers $\boldsymbol{C_k} = [\boldsymbol{l_k}, \boldsymbol{a_k}, \boldsymbol{b_k}, \boldsymbol{x_k}, \boldsymbol{y_k}]^{\boldsymbol{T}}$ on a regular grid spaced $S = \sqrt{\frac{N}{K}}$ pixels apart.
   - each superpixel has approximately $\frac{N}{K}$ pixels



$$\frac{N}{K} = \frac{1600}{16} = 100$$

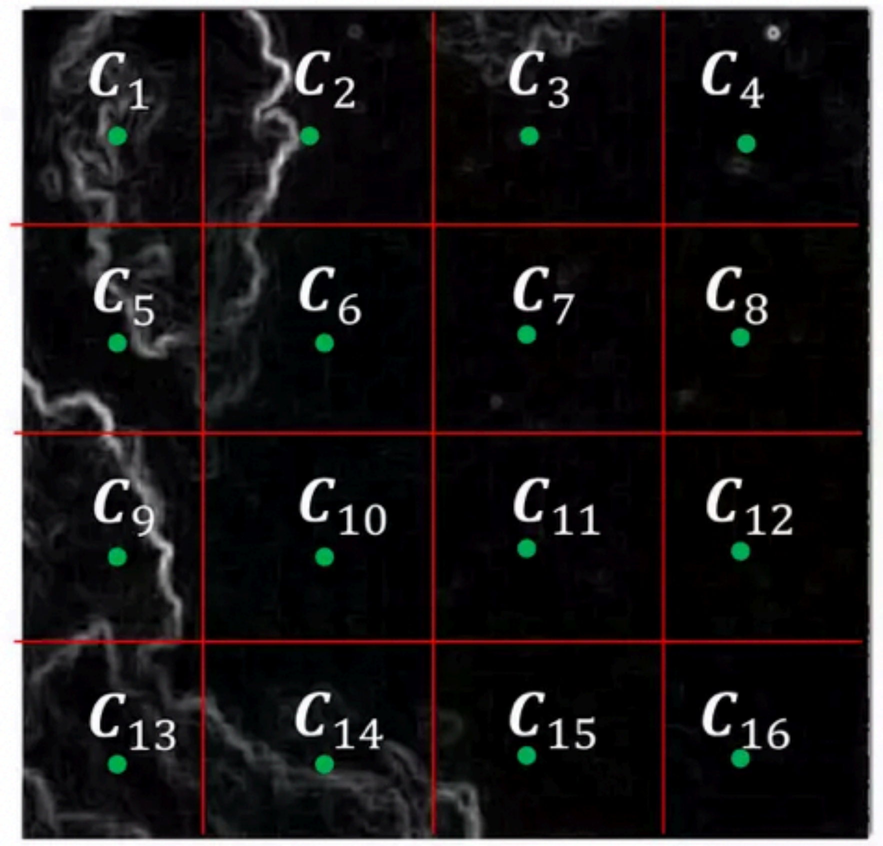
$$S = \sqrt{\frac{N}{K}} = \sqrt{100} = 10$$

https://www.slideshare.net/ShimaFoolad1/superpixel-algorithms-whatershed-meanshift-slic-bslic-foolad

SLIC Segmentation

2. move these cluster centers to the positions with the lowest gradients in a 3 × 3 neighborhood;

This is done to avoid centering a superpixel on an edge, and to reduce to chance of seeding superpixel with a noisy pixel.

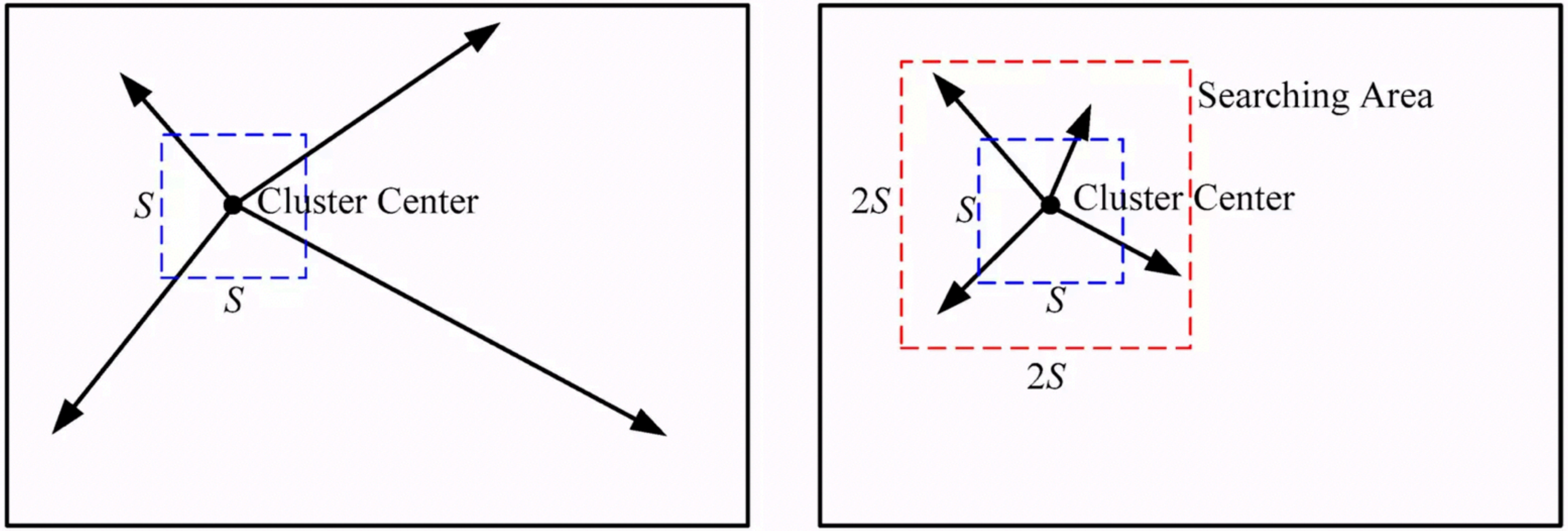


$$G(x,y) = \|I(x,y-1) - I(x,y+1)\|^2 + \|I(x-1,y) - I(x+1,y)\|^2$$

https://www.slideshare.net/ShimaFoolad1/superpixel-algorithms-whatershed-meanshift-slic-bslic-foolad

SLIC Segmentation

3. Assign pixels. Designate each pixel to a closest cluster center in a local search space;



(a) K-means algorithm

(b) SLIC algorithm

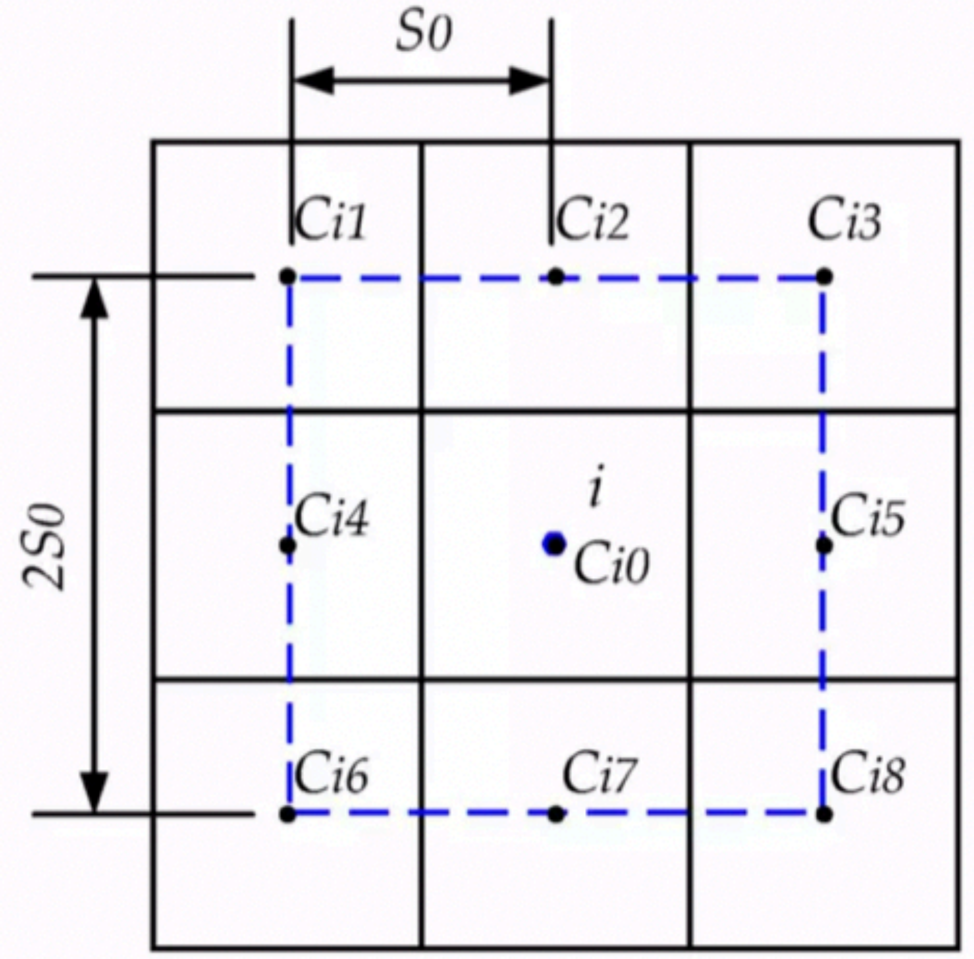
SLIC Segmentation

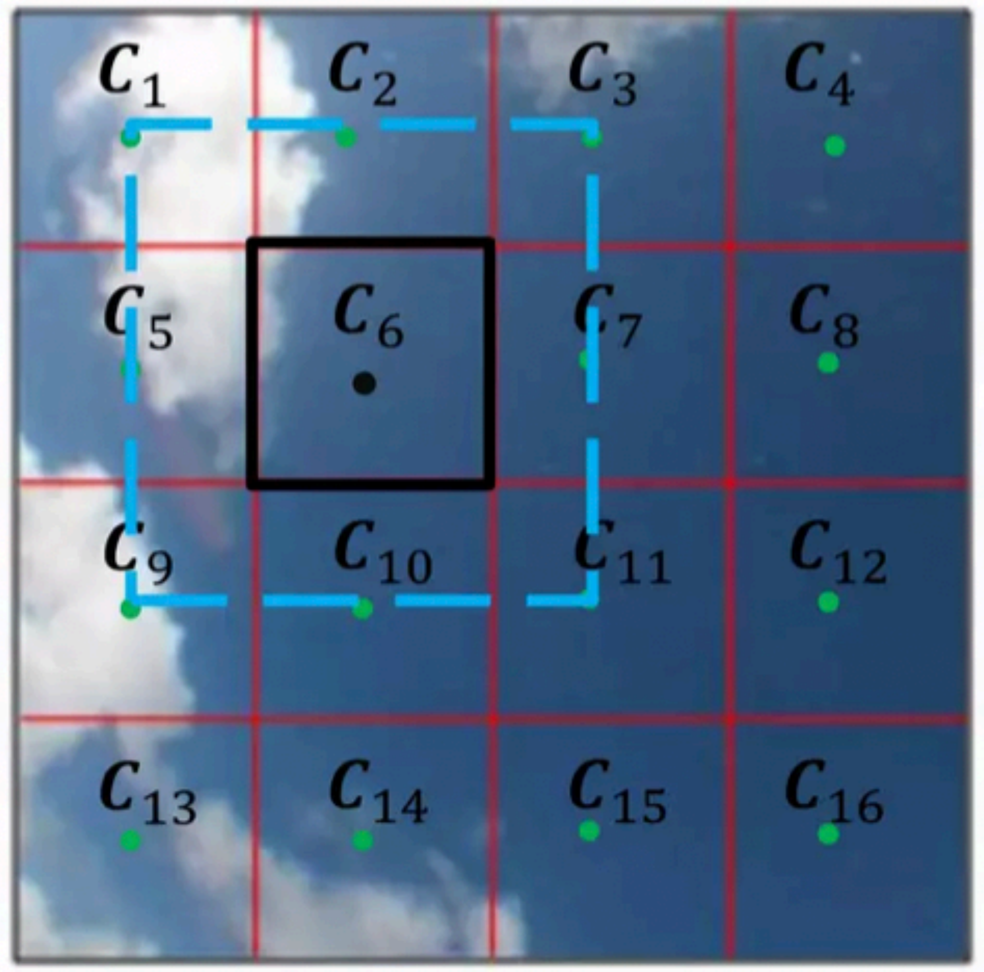
3. Assign pixels. Designate each pixel to a closest cluster center in a local search space;

for each cluster center C_k

Assign the best matching pixels from a $2S \times 2S$ around the cluster center according to the distance measure.

end for





: Superpixel : $2S0 \times 2S0$ search region • : Superpixel center • : Image pixel

https://www.slideshare.net/ShimaFoolad1/superpixel-algorithms-whatershed-meanshift-slic-bslic-foolad

SLIC Segmentation

3. Assign pixels. Designate each pixel to a closest cluster center in a local search space;

Distance measure

Color distance: $$d_c = \sqrt{\left(l_j - l_i\right)^2 + \left(a_j - a_i\right)^2 + \left(b_j - b_i\right)^2}$$

position distance: $$d_s = \sqrt{\left(x_j - x_i\right)^2 + \left(y_j - y_i\right)^2}$$

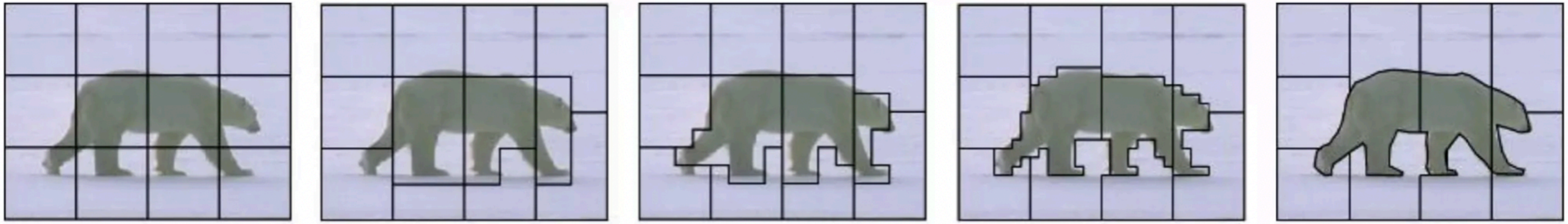
$$D' = \sqrt{\left(\frac{d_c}{N_c}\right)^2 + \left(\frac{d_s}{N_s}\right)^2} \qquad D' = \sqrt{\left(\frac{d_c}{m}\right)^2 + \left(\frac{d_s}{S}\right)^2} \qquad D = \sqrt{d_c^{\,2} + \left(\frac{d_s}{S}\right)^2 m^2}$$

m= Compactness of superpixel $\quad S = \sqrt{N / K}$

https://www.slideshare.net/ShimaFoolad1/superpixel-algorithms-whatershed-meanshift-slic-bslic-foolad

SLIC Segmentation

4. Update cluster centers. Set each cluster center as the mean of all pixels in the corresponding cluster;
5. Repeat steps (3)–(4) until the clusters do not change or error (difference between previous cluster and new cluster) is converge.



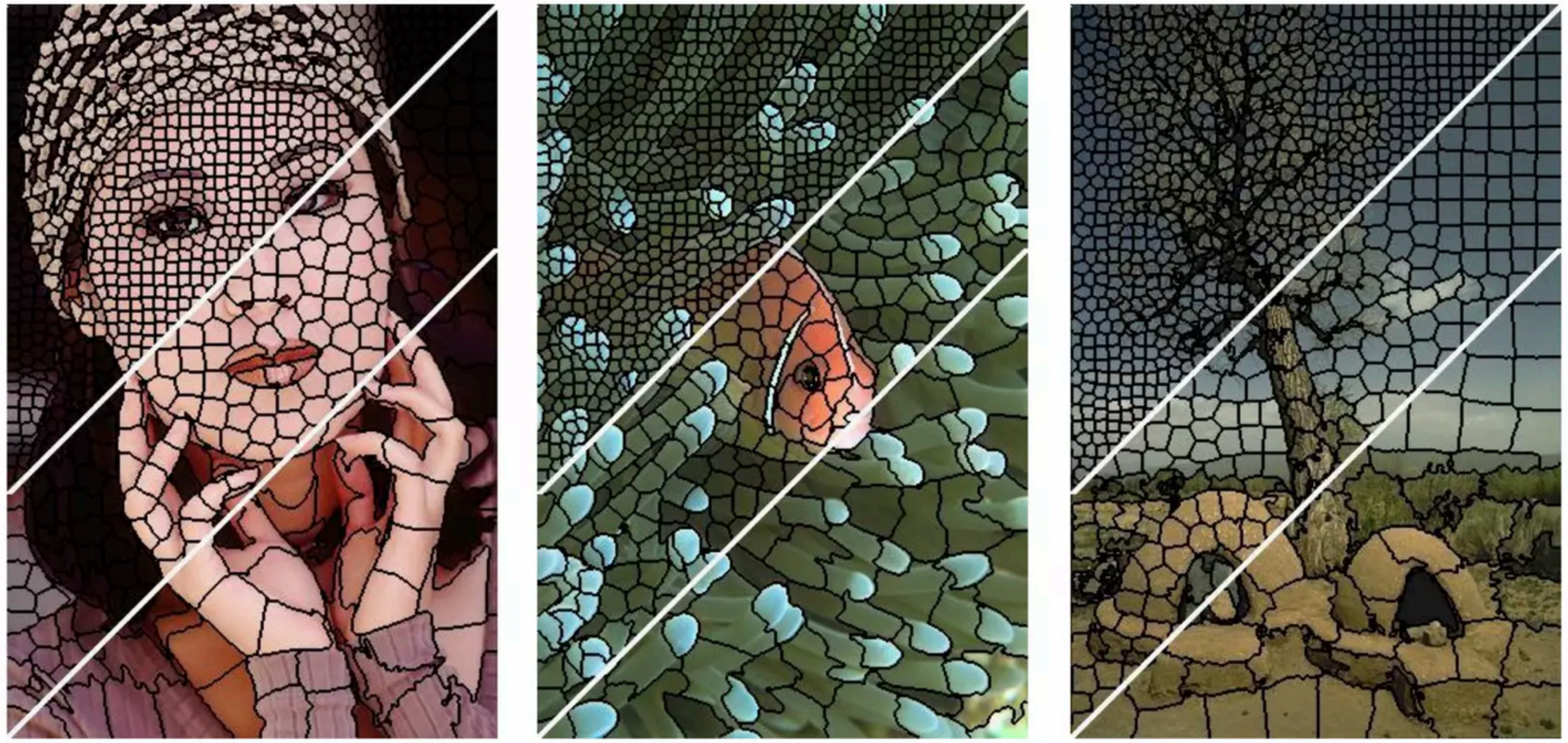
initialization iteration 1 iteration 2 iteration 3 iteration 4

https://www.slideshare.net/ShimaFoolad1/superpixel-algorithms-whatershed-meanshift-slic-bslic-foolad

SLIC Segmentation

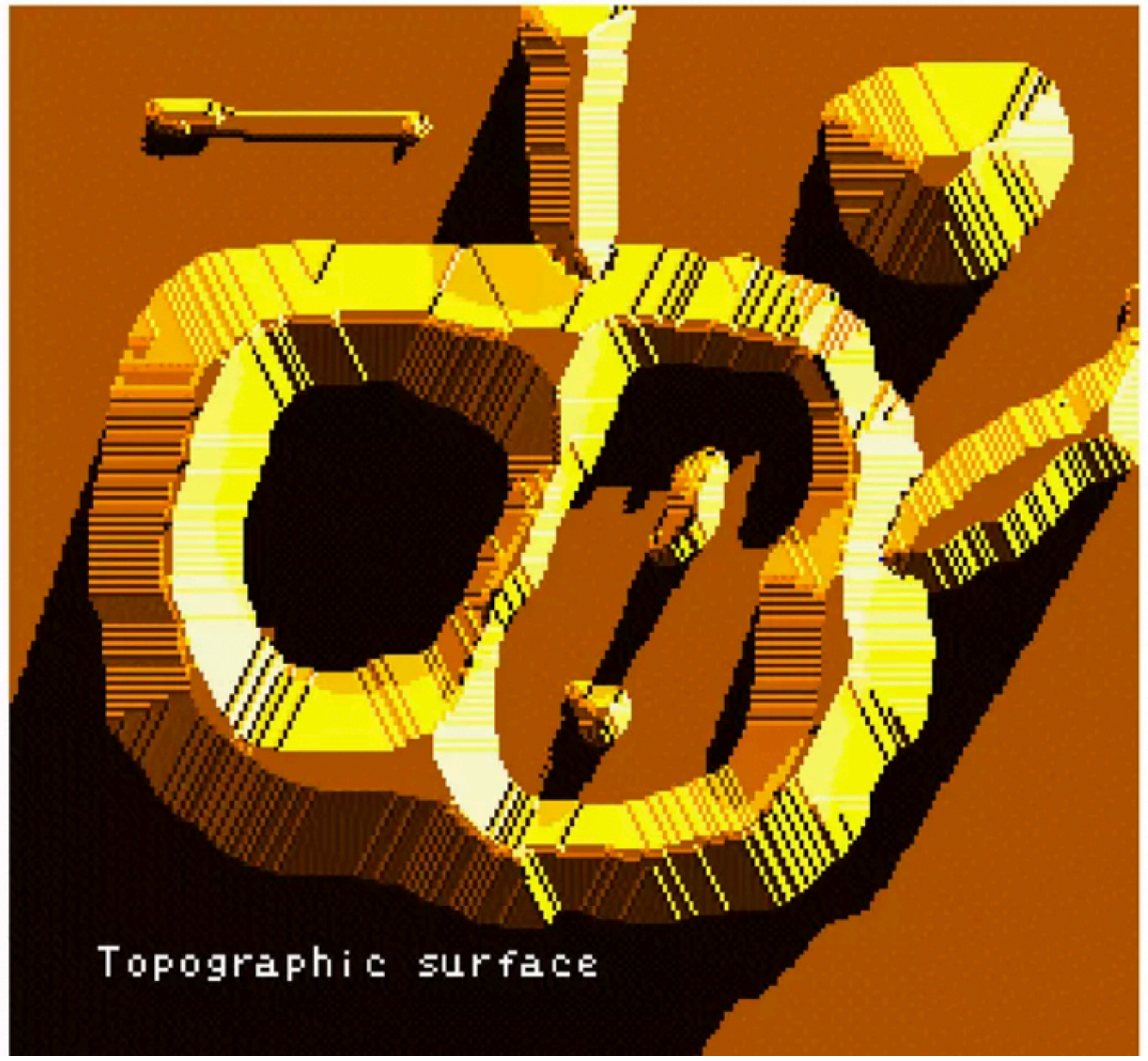
Simple Linear Iterative Clustering (SLIC) algorithm

Sample segmentation output



Watershed Segmentation

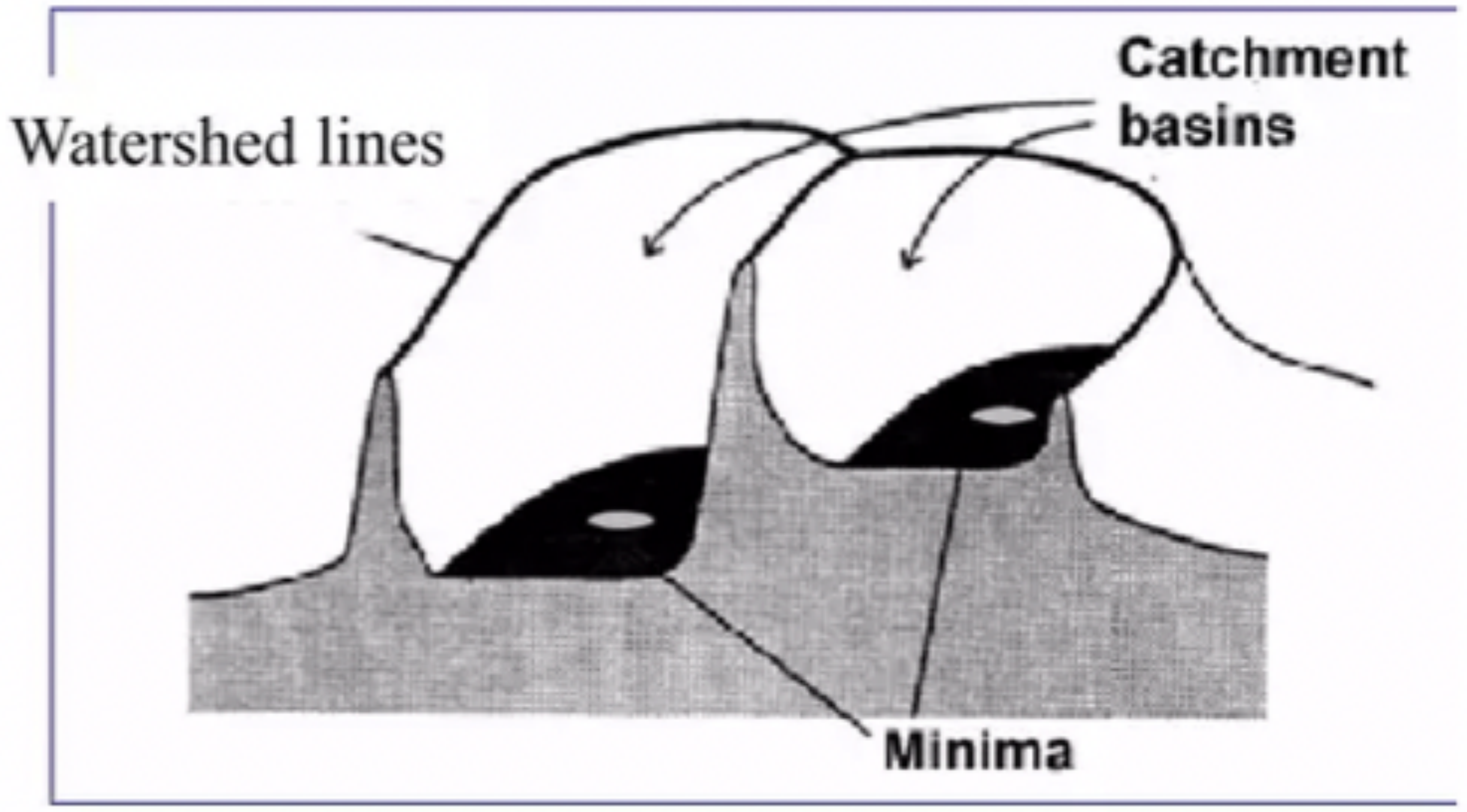
- **Image is visualized in 3-DIMENSIONS.**
  - 2 spatial dimensions
  - grey levels
- Any grey tone image can be considered as a **TOPOLOGICAL SURFACE.**



https://www.slideshare.net/SaHussain1/watershed-segmentation-image-processing

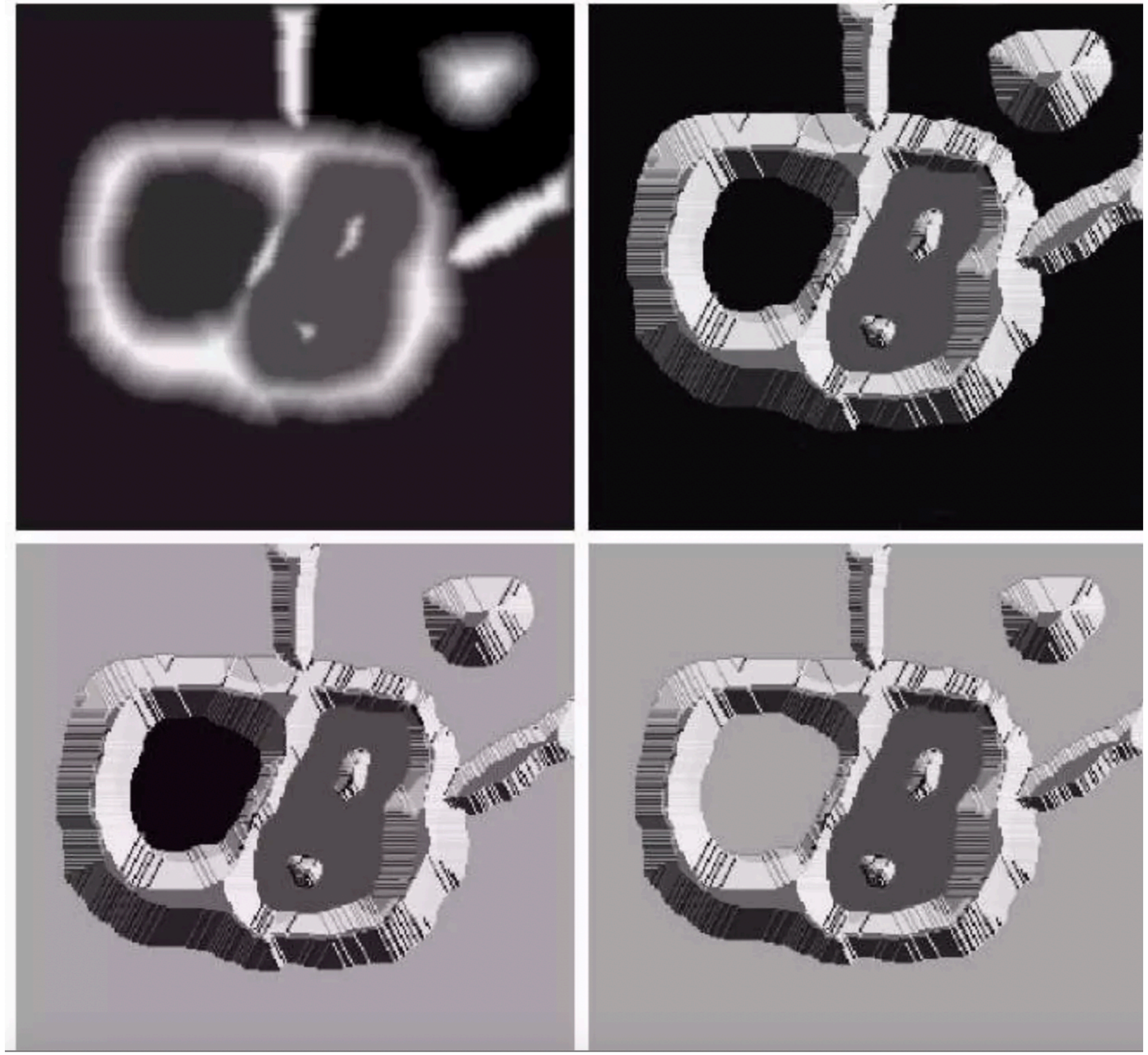
Watershed Segmentation

- **Punch** the regional minimum and flood the entire topography at uniform rate from below
- A **dam is built** to prevent the rising water from distinct catchment basins from merging
- Eventually only the tops of the dams are visible above the water line
- These dam boundaries correspond to the **divide lines** of the watersheds



- In order to prevent water from **spilling** out of the structure we imagine the entire topography to be enclosed by **dams of height greater than highest possible mountain**
- The value of **the height is determined by the highest** possible gray-level value in the input image

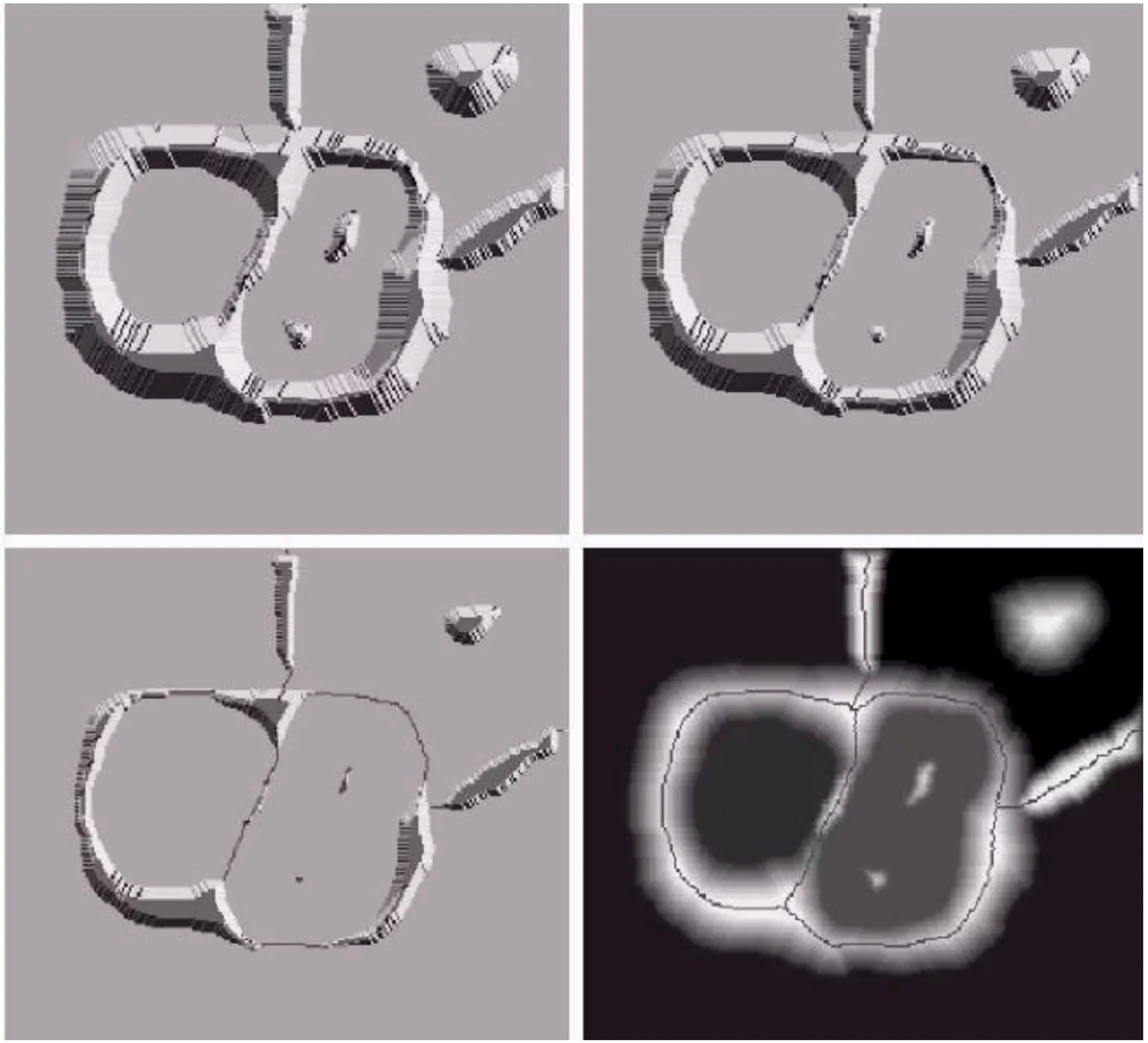
Watershed Segmentation



https://www.slideshare.net/SaHussain1/watershed-segmentation-image-processing

Watershed Segmentation

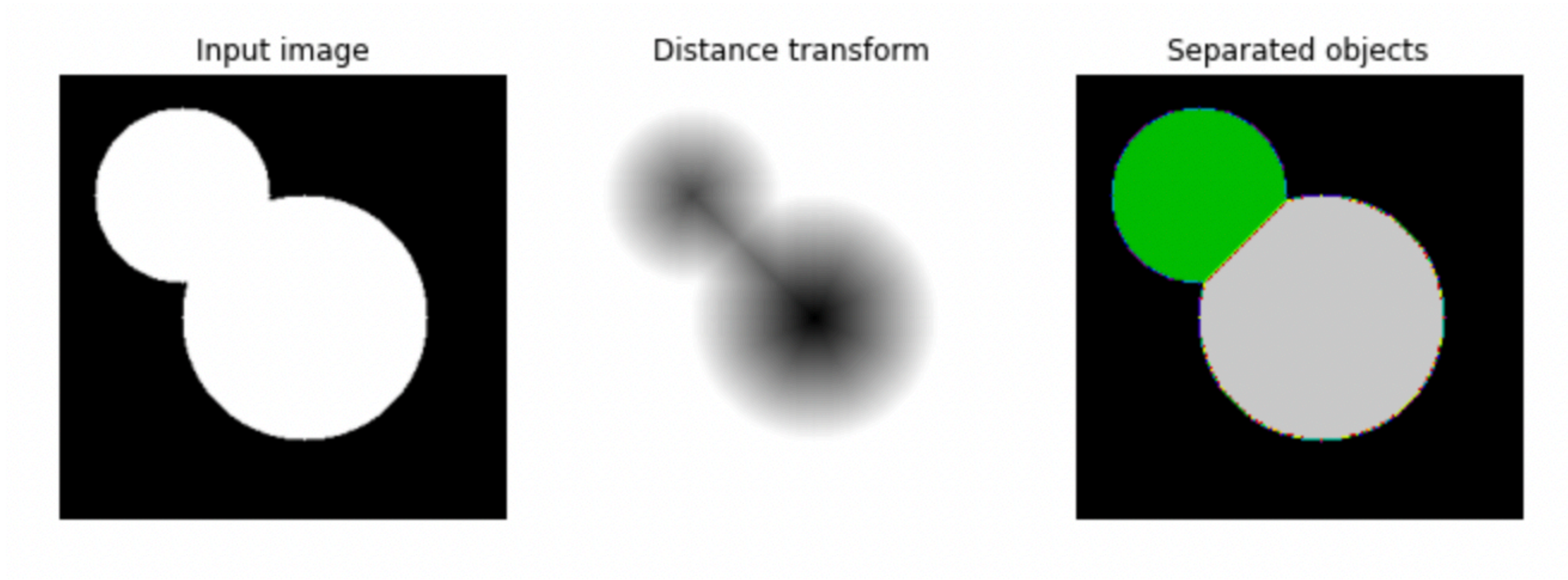
The content is added from different sources



https://www.slideshare.net/SaHussain1/watershed-segmentation-image-processing

Watershed Segmentation

The content is added from different sources

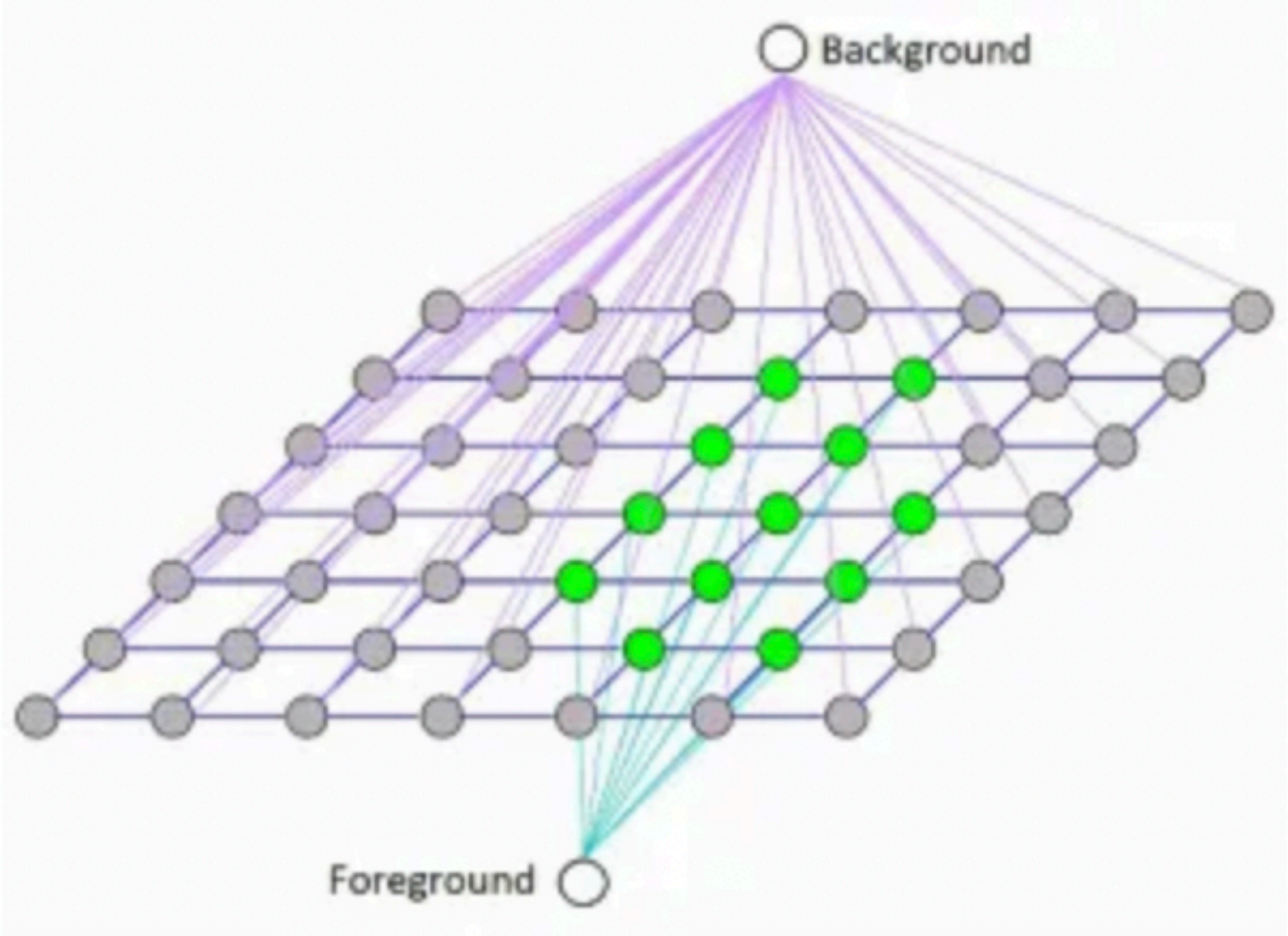


https://datahacker.rs/007-opencv-projects-image-segmentation-with-watershed-algorithm/

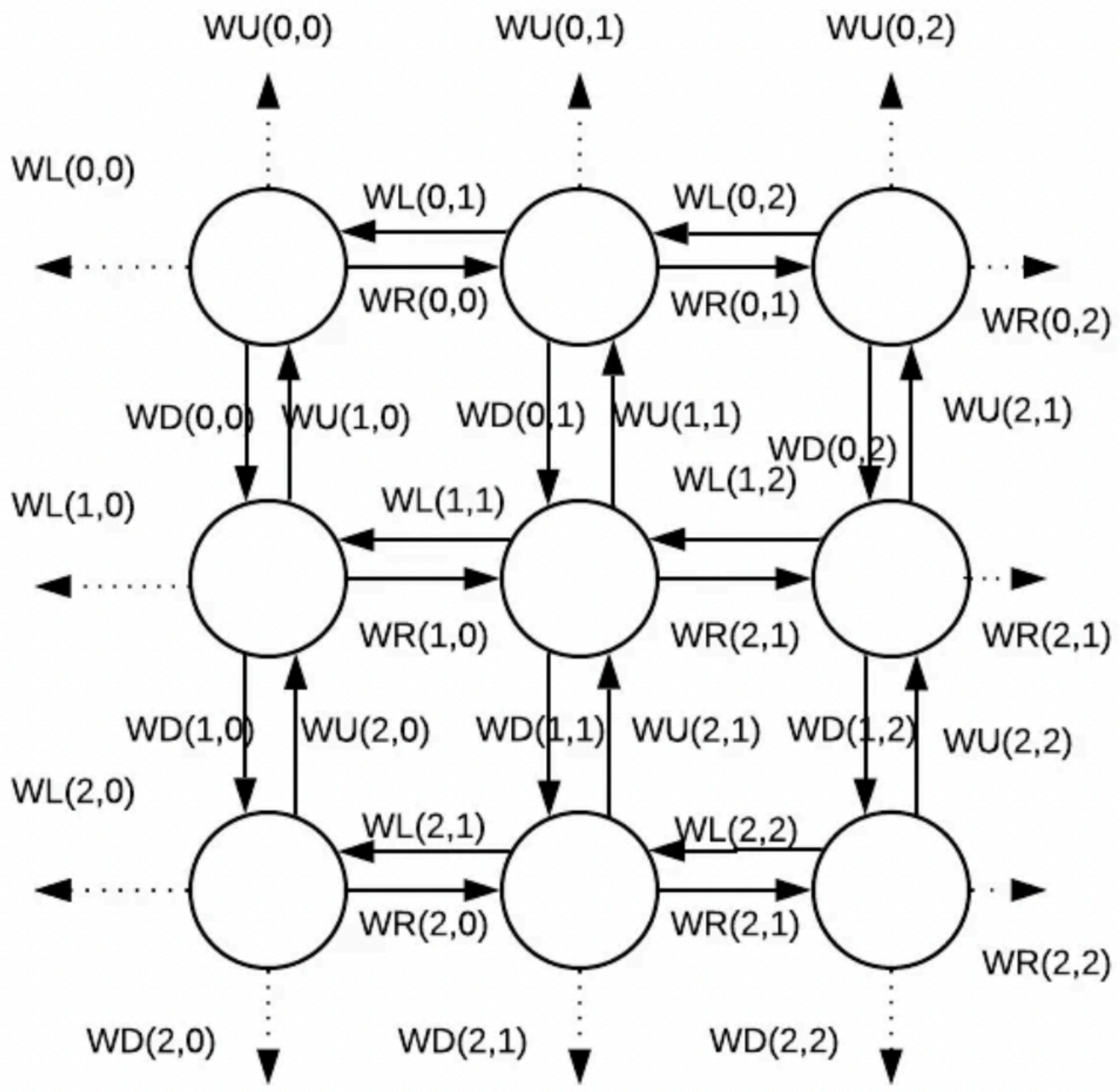
GrabCut Segmentation

You start off by drawing a rectangle on an image and this rectangle should include the subjects of the image, like a person or a dog. Subsequently, the area lying outside of the rectangle you've just drawn is automatically defined as the background. The data contained in the background is used as a reference to distinguish background areas from foreground areas within the defined rectangle.

To make it simple, this algorithm defines the area in the rectangle as a color distribution model using the Gaussian Mixture Model(GMM), where each pixel will be given a label to tell whether it is a foreground, background, or unknown. If you understand a little about image processing, every pixel is connected to one another by a gradient and so this model will encourage neighbouring pixels of similar color distribution to have the exact same label.



https://medium.com/analytics-vidhya/computer-vision-understanding-grabcut-algorithm-without-the-maths-9a97ef4c5ba3



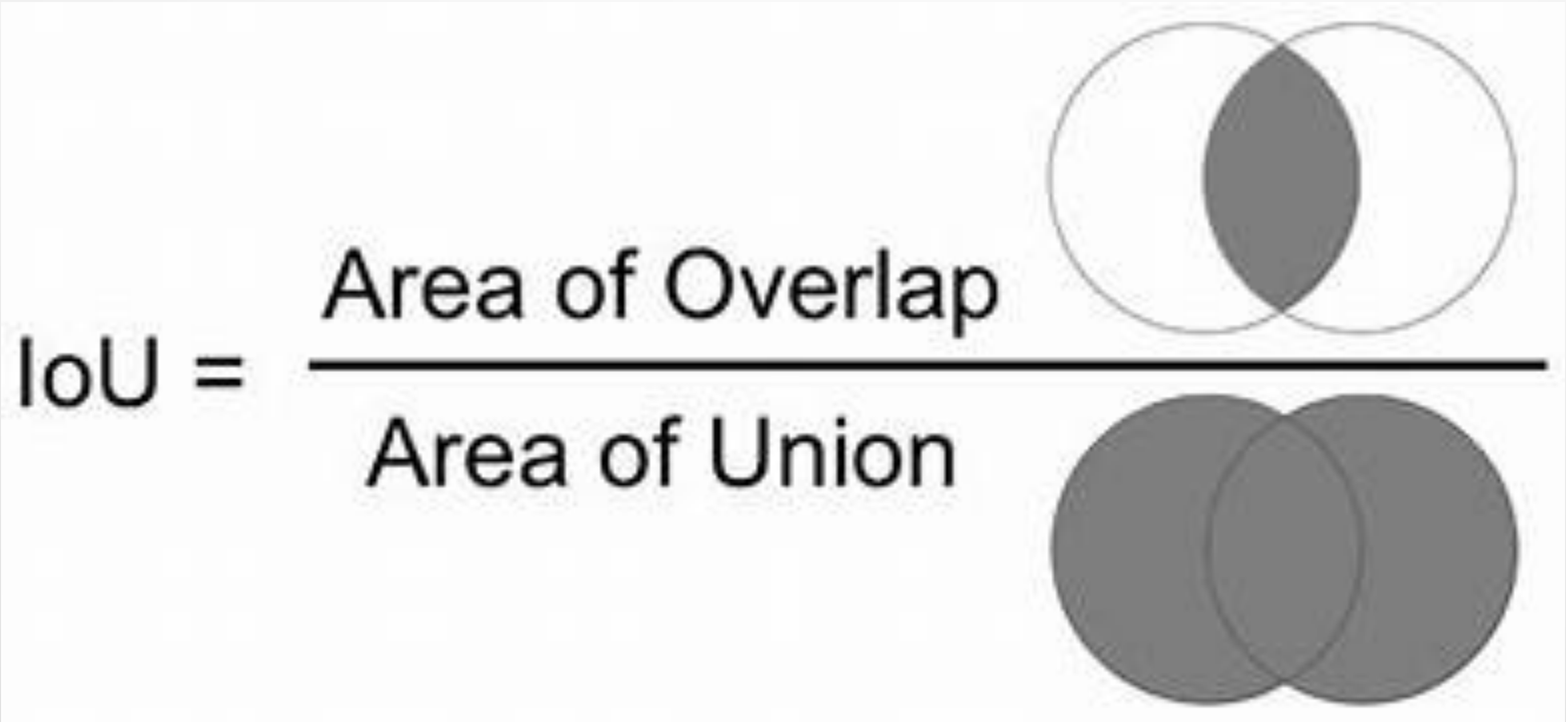
1. A sharp shift of pixel intensity from background to foreground
2. Foreground parts don’t tangle with background parts too much
3. Image is not too blurry

https://medium.com/analytics-vidhya/image-segmentation-with-yolov3-and-grabcut-59a0abaafa3e

Segmentation Metrics

Pixel accuracy is perhaps the easiest to understand conceptually. It is the percent of pixels in your image that are classified correctly.

Intersection-Over-Union



Metric

These are precision and recall. While precision gives the proportion of positive predictions which are truly positive, recall gives the proportion of TP cases over all actually positive cases.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

Precision indicates the reliability of a model in predicting a class of interest. When the model is related to win / loss prediction of cricket, precision indicates how often it predicts the win correctly. ratio of the correctly +ve labeled by our program to all +ve labeled.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} = \frac{85}{85 + 4} = \frac{85}{89} = 95.5\%$$

Recall indicates the proportion of correct prediction of positives to the total number of positives. Recall is the ratio of the correctly +ve labeled by our program to all who have won match in reality. Of all the people who have BP, how many of those we correctly predict?

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} = \frac{85}{85 + 2} = \frac{85}{87} = 97.7\%$$

F-measure is another measure of model performance which combines the precision and recall. It takes the harmonic mean of precision and recall

$$F\text{-measure} = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

$$F\text{-measure} = \frac{2 \times 0.955 \times 0.977}{0.955 + 0.977} = \frac{1.866}{1.932} = 96.6\%$$

Metric

		Actual Values	
		Positive (1)	Negative (0)
Predicted Values	Positive (1)	TP	FP
	Negative (0)	FN	TN

		ACTUAL VALUES	
		POSITIVE	NEGATIVE
PREDICTED VALUES	POSITIVE	560	60
	NEGATIVE	50	330

- True Positive (TP) = 560; meaning 560 positive class data points were correctly classified by the model
- True Negative (TN) = 330; meaning 330 negative class data points were correctly classified by the model
- False Positive (FP) = 60; meaning 60 negative class data points were incorrectly classified as belonging to the positive class by the model
- False Negative (FN) = 50; meaning 50 positive class data points were incorrectly classified as belonging to the negative class by the model

Precision summarizes the fraction of examples assigned the positive class that belong to the positive class.

Precision = TruePositive / (TruePositive + FalsePositive)

Recall summarizes how well the positive class was predicted and is the same calculation as sensitivity.

Recall = TruePositive / (TruePositive + FalseNegative)

Precision and recall can be combined into a single score that seeks to balance both concerns, called the F-score or the F-measure.

F-Measure = (2 * Precision * Recall) / (Precision + Recall)
The F-Measure is a popular metric for imbalanced classification.