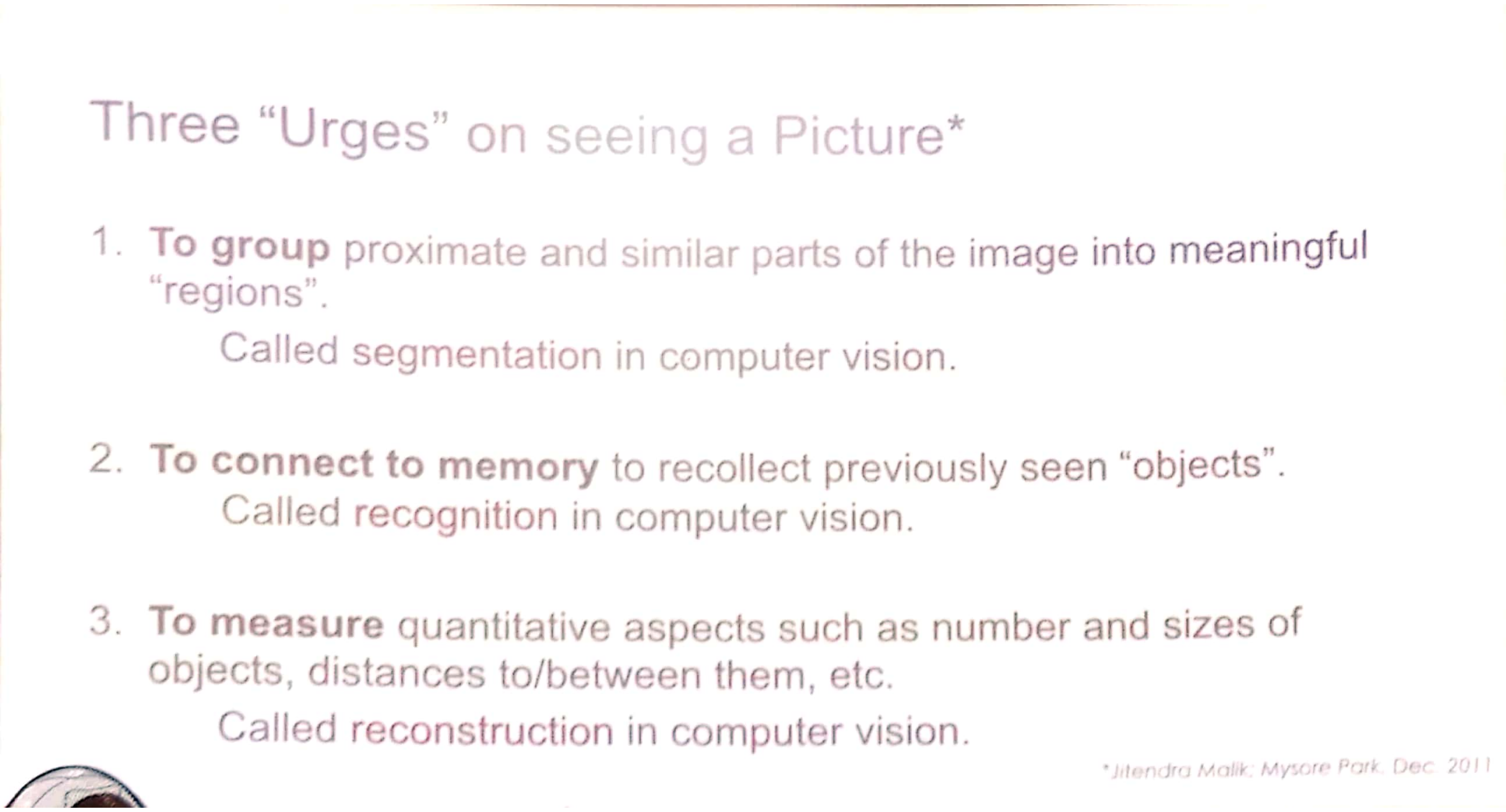


Three “Urges” on seeing a Picture*

1. **To group** proximate and similar parts of the image into meaningful “regions”.

   Called segmentation in computer vision.

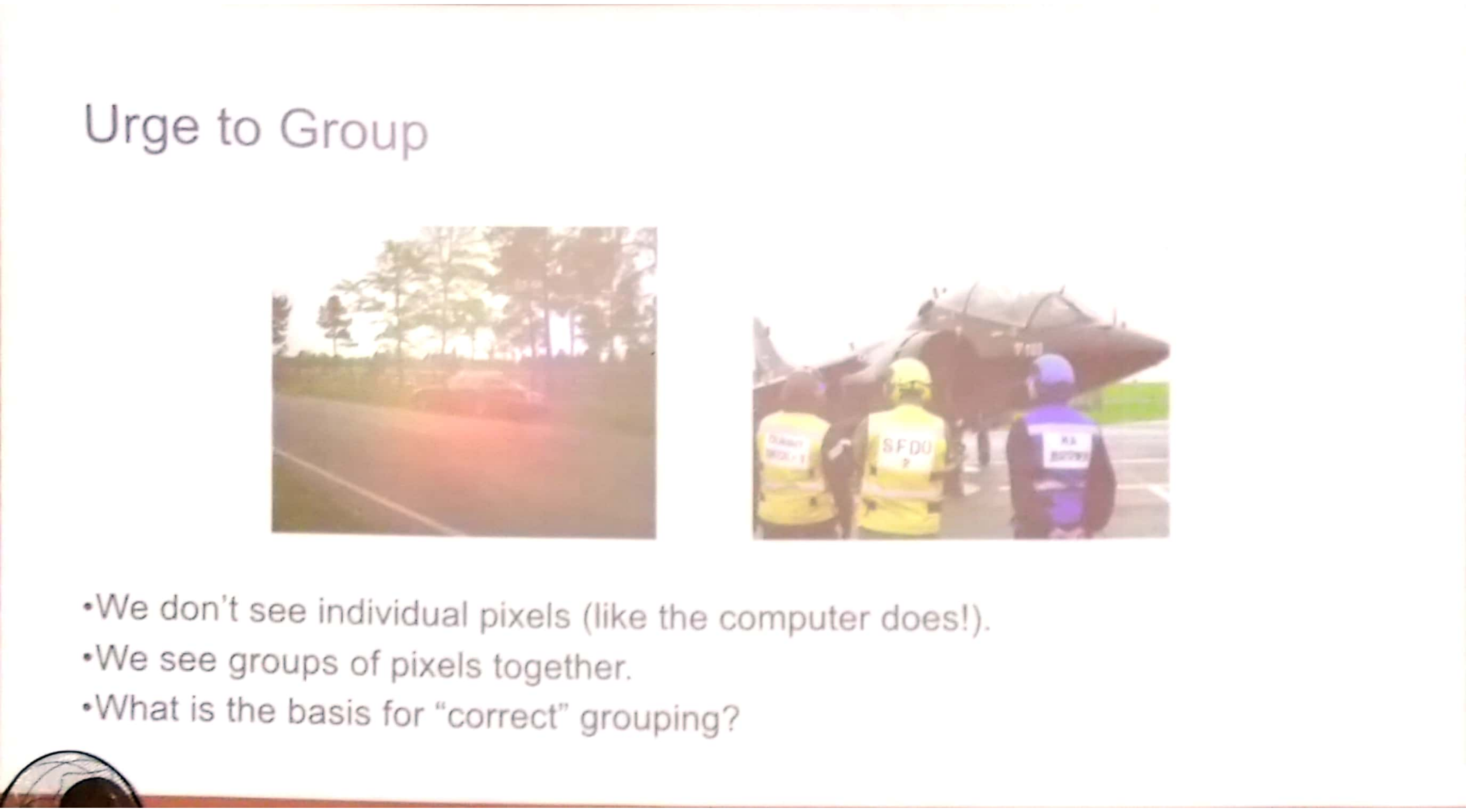
2. **To connect to memory** to recollect previously seen “objects”.

   Called recognition in computer vision.

3. **To measure** quantitative aspects such as number and sizes of objects, distances to/between them, etc.

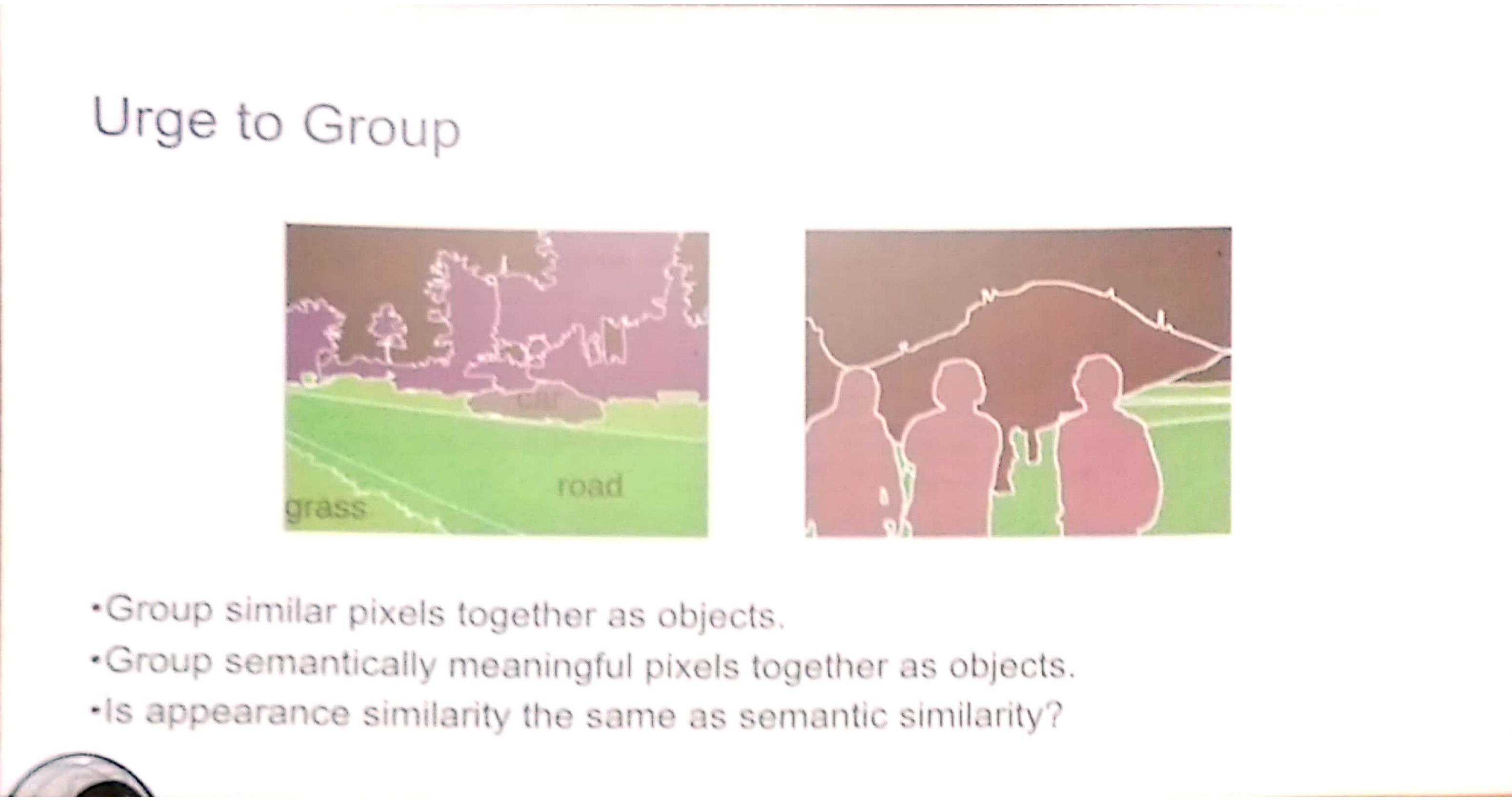
   Called reconstruction in computer vision.

*Jitendra Malik; Mysore Park, Dec. 2011



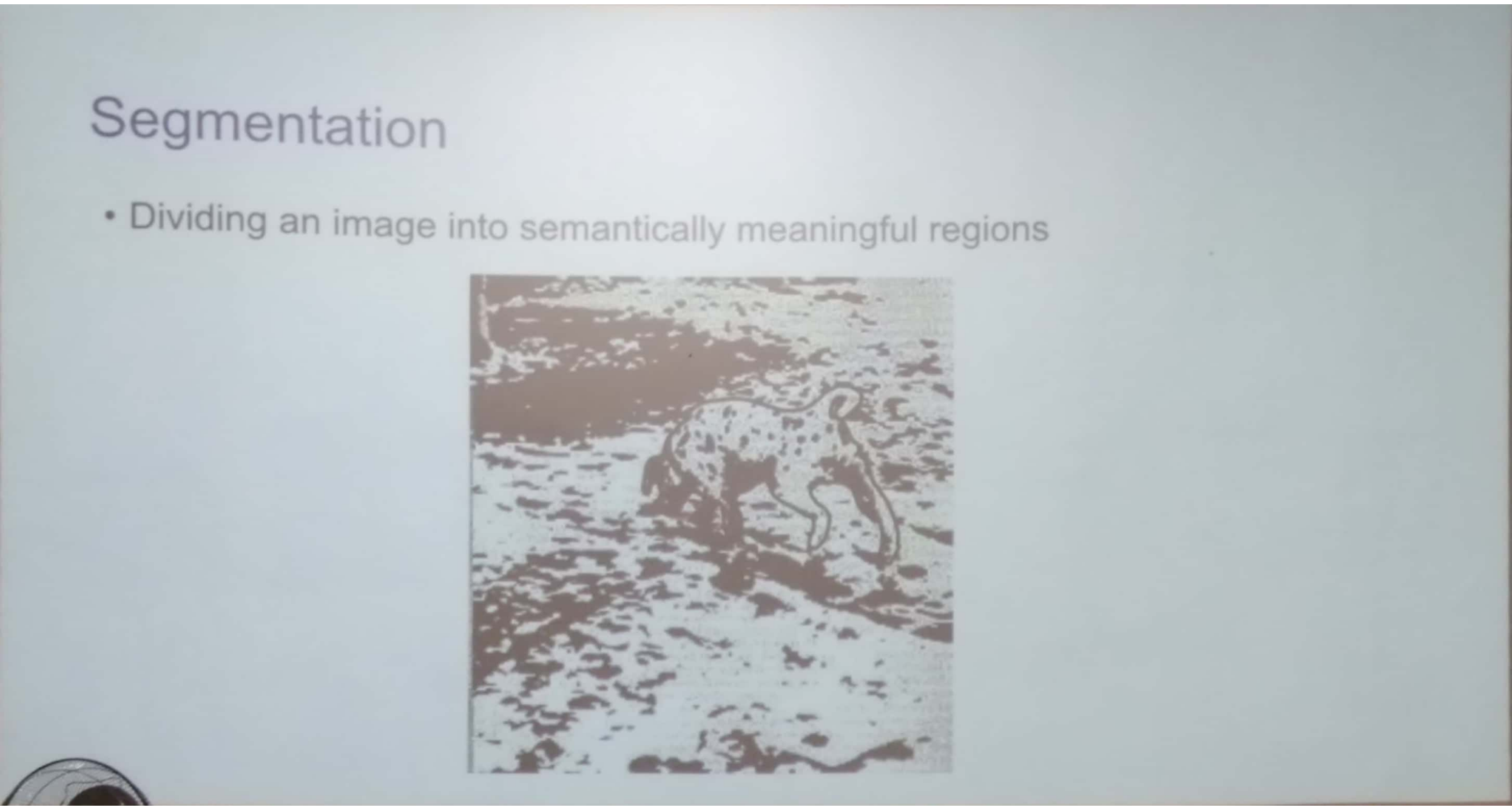
Urge to Group

- We don't see individual pixels (like the computer does!).
- We see groups of pixels together.
- What is the basis for "correct" grouping?



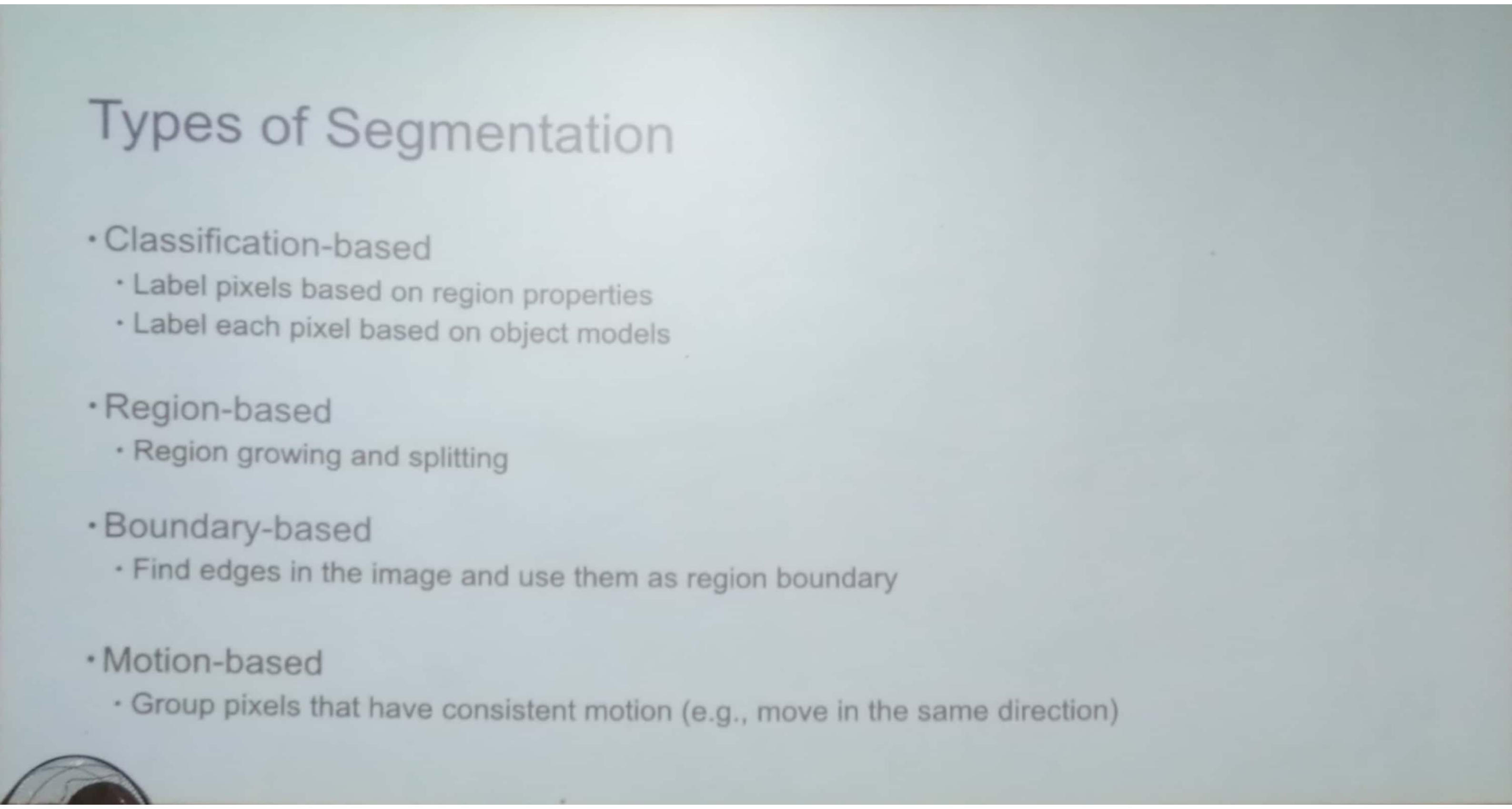
Urge to Group

- Group similar pixels together as objects.
- Group semantically meaningful pixels together as objects.
- Is appearance similarity the same as semantic similarity?



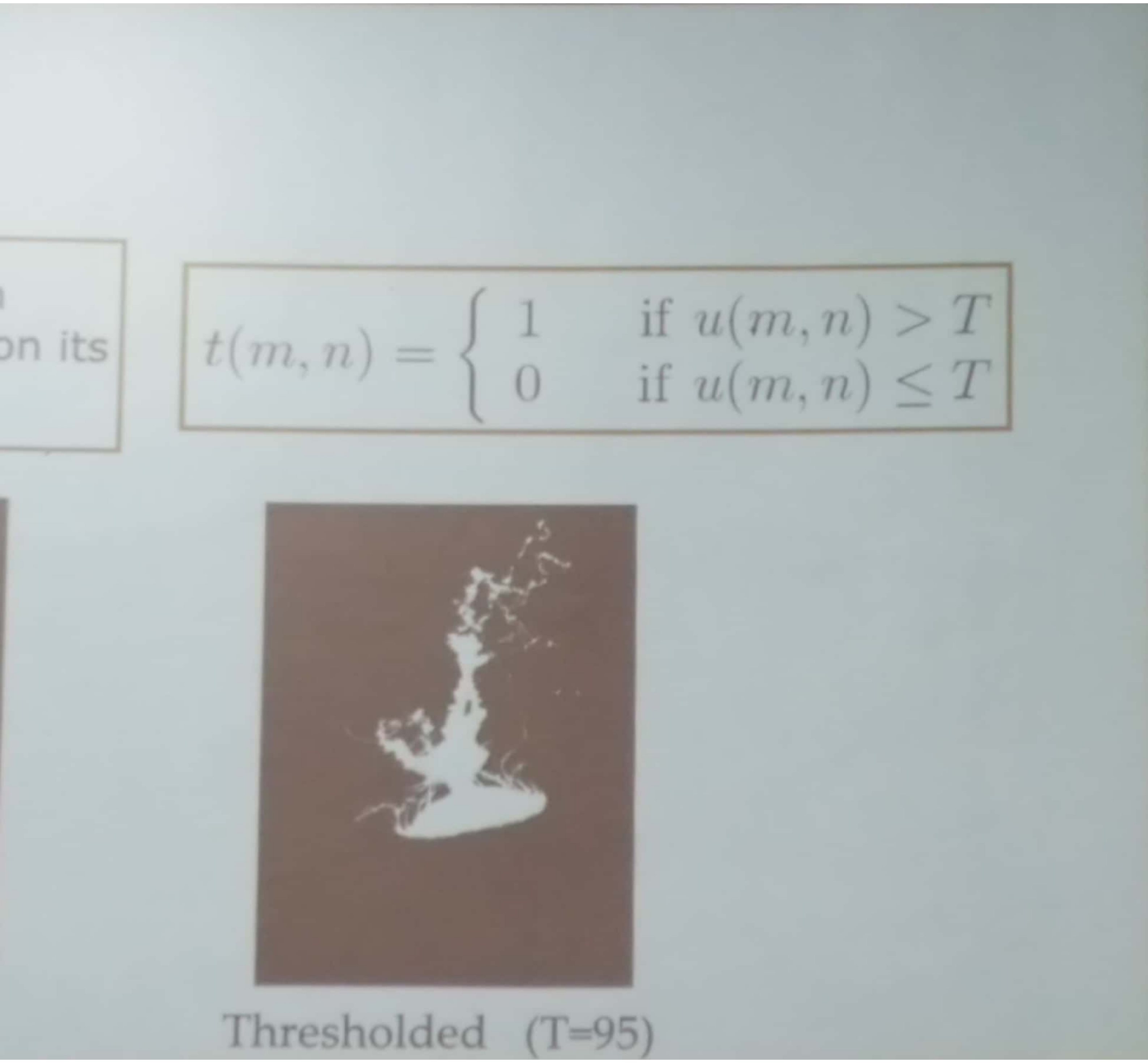
Segmentation

- Dividing an image into semantically meaningful regions



Types of Segmentation

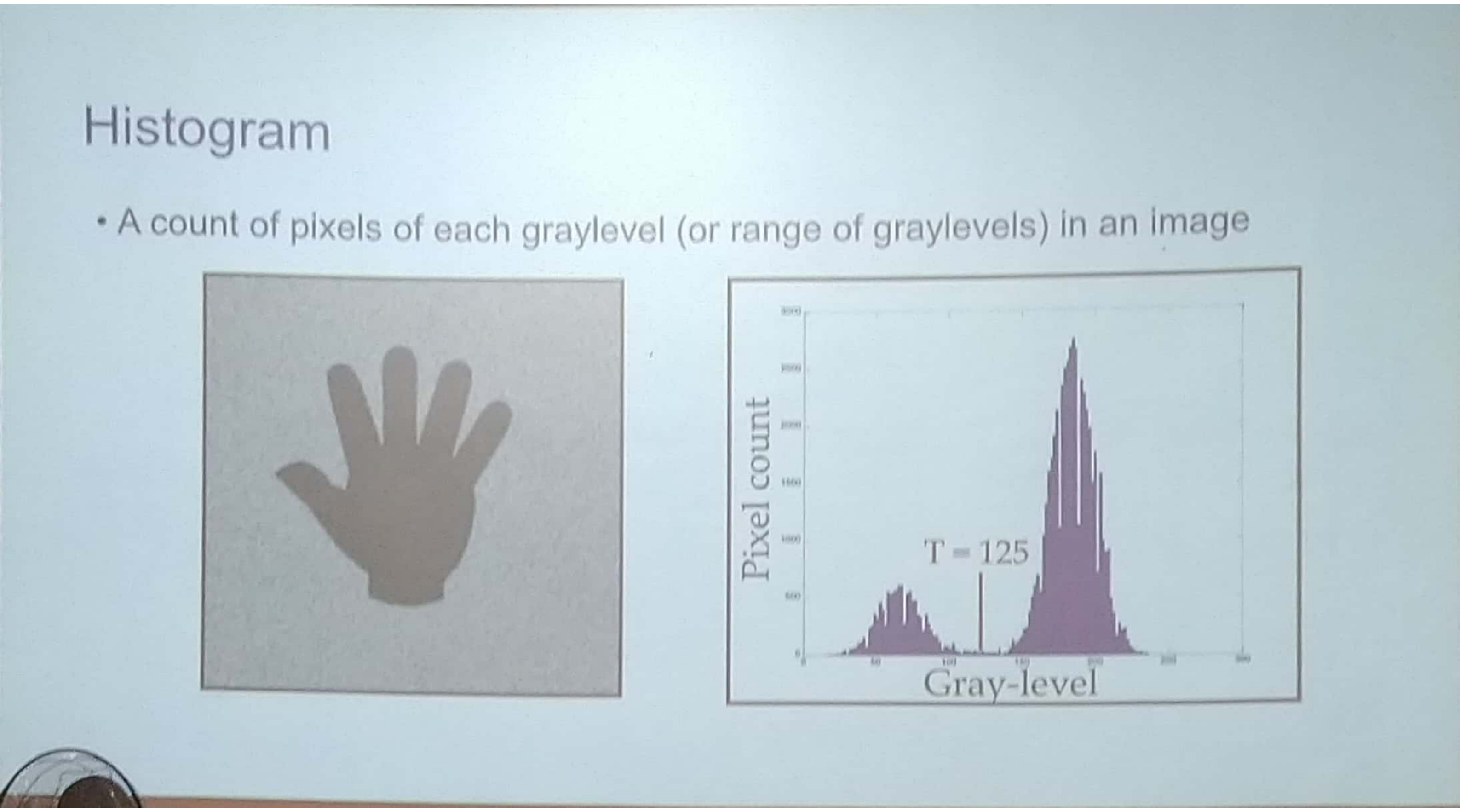
- Classification-based
  - Label pixels based on region properties
  - Label each pixel based on object models
- Region-based
  - Region growing and splitting
- Boundary-based
  - Find edges in the image and use them as region boundary
- Motion-based
  - Group pixels that have consistent motion (e.g., move in the same direction)



on its

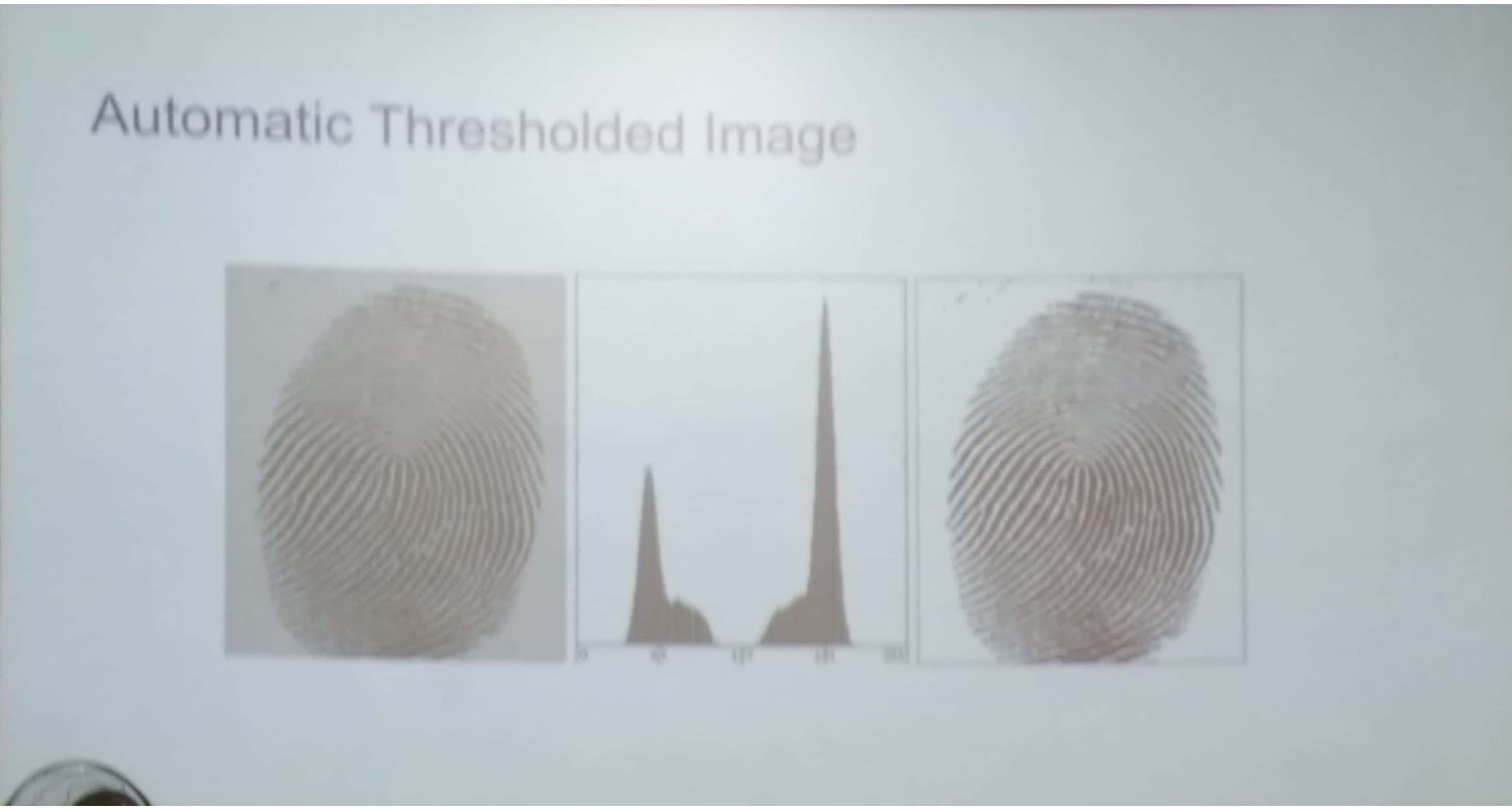
$$t(m,n) = \begin{cases} 1 & \text{if } u(m,n) > T \\ 0 & \text{if } u(m,n) \le T \end{cases}$$

Thresholded (T=95)

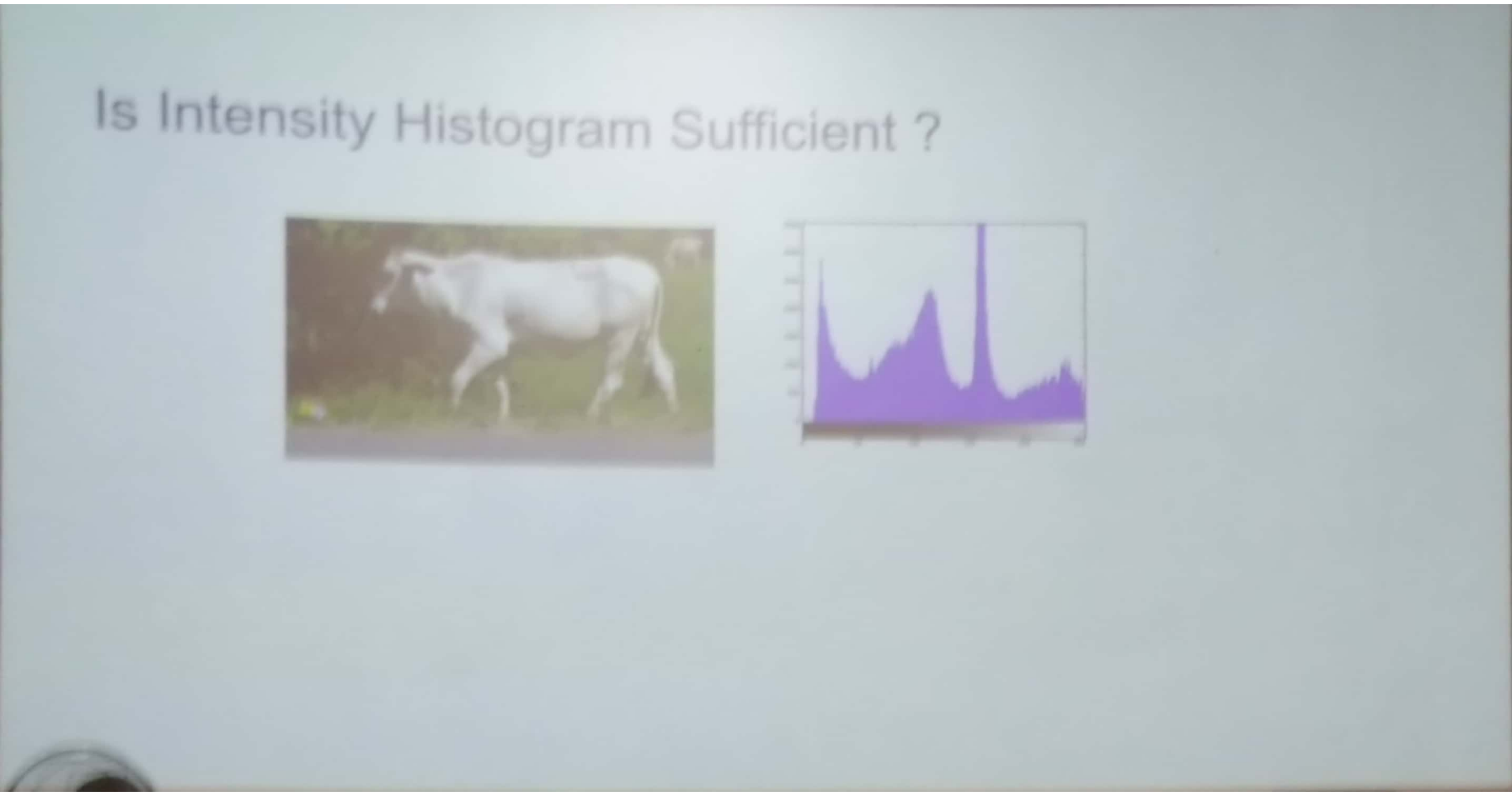


Histogram

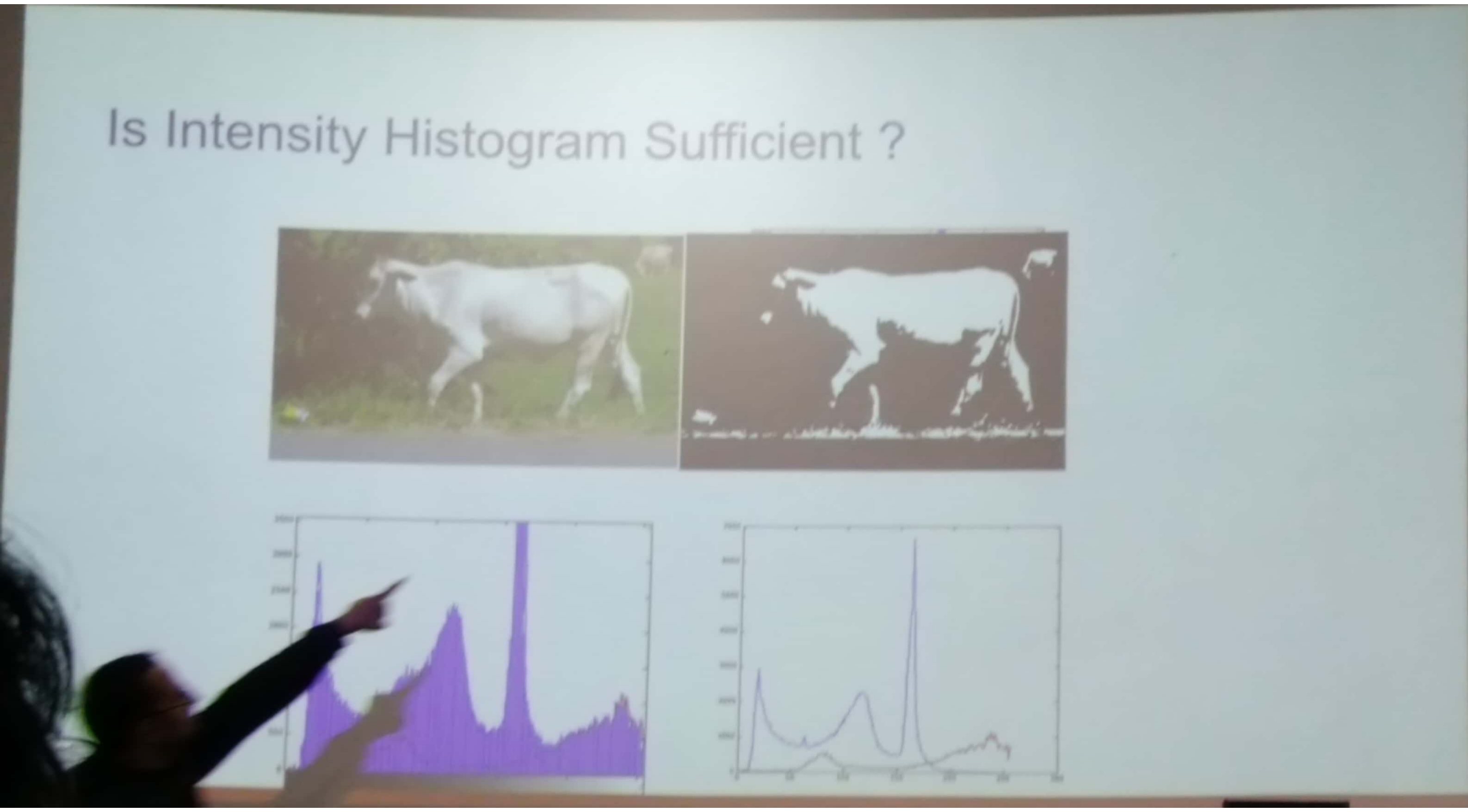
- A count of pixels of each graylevel (or range of graylevels) in an image



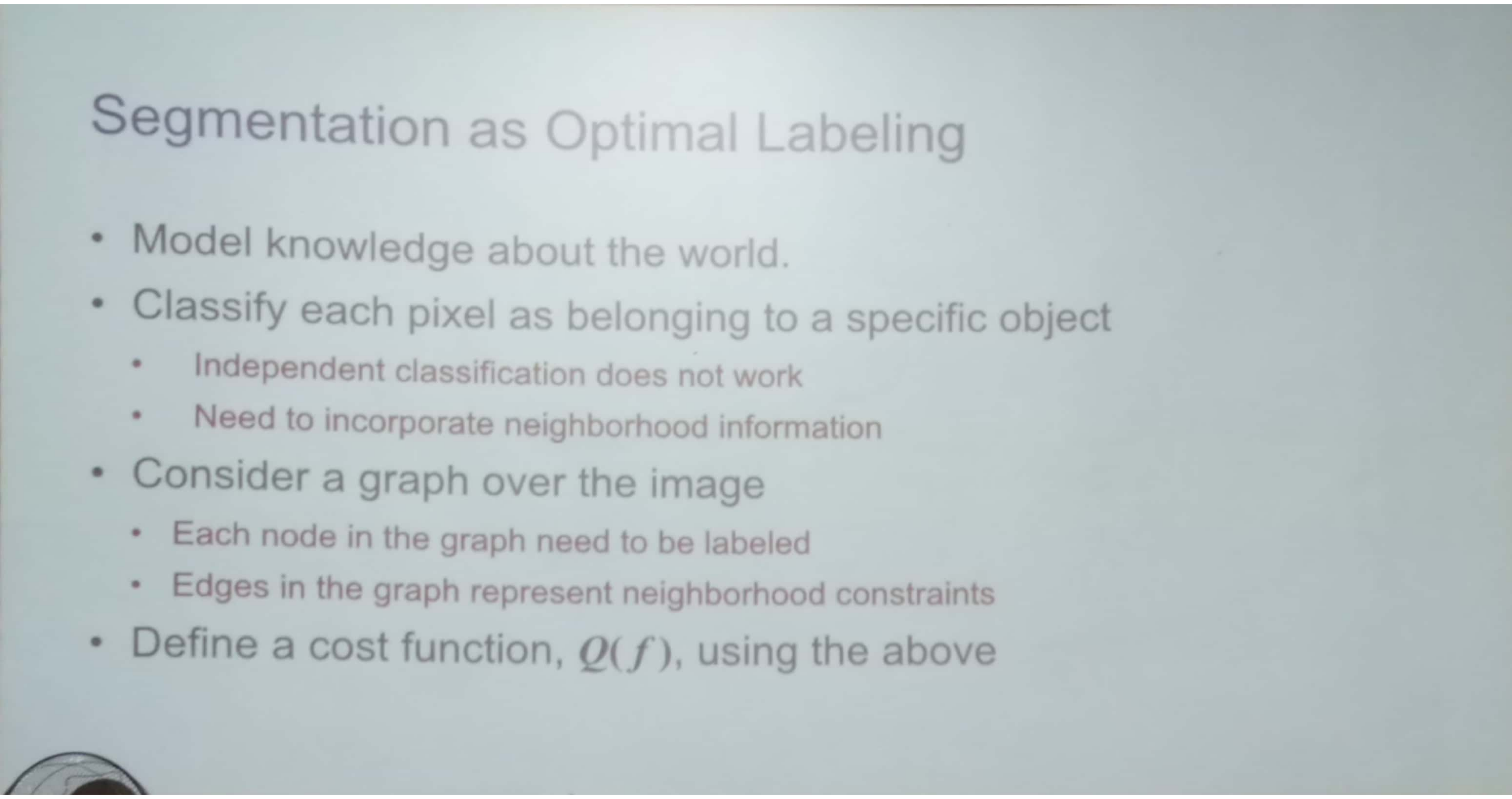
Automatic Thresholded Image



Is Intensity Histogram Sufficient ?

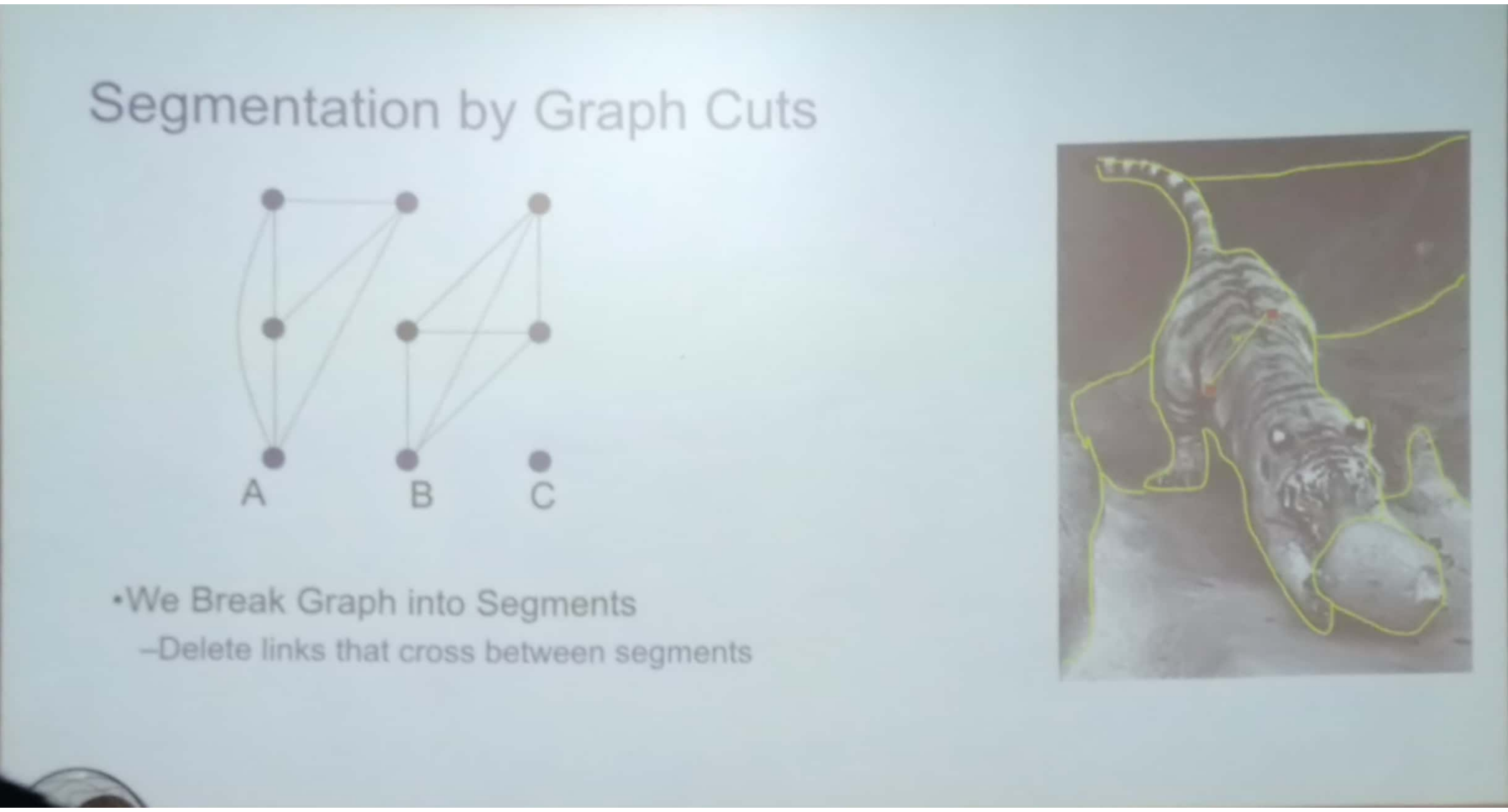


Is Intensity Histogram Sufficient ?



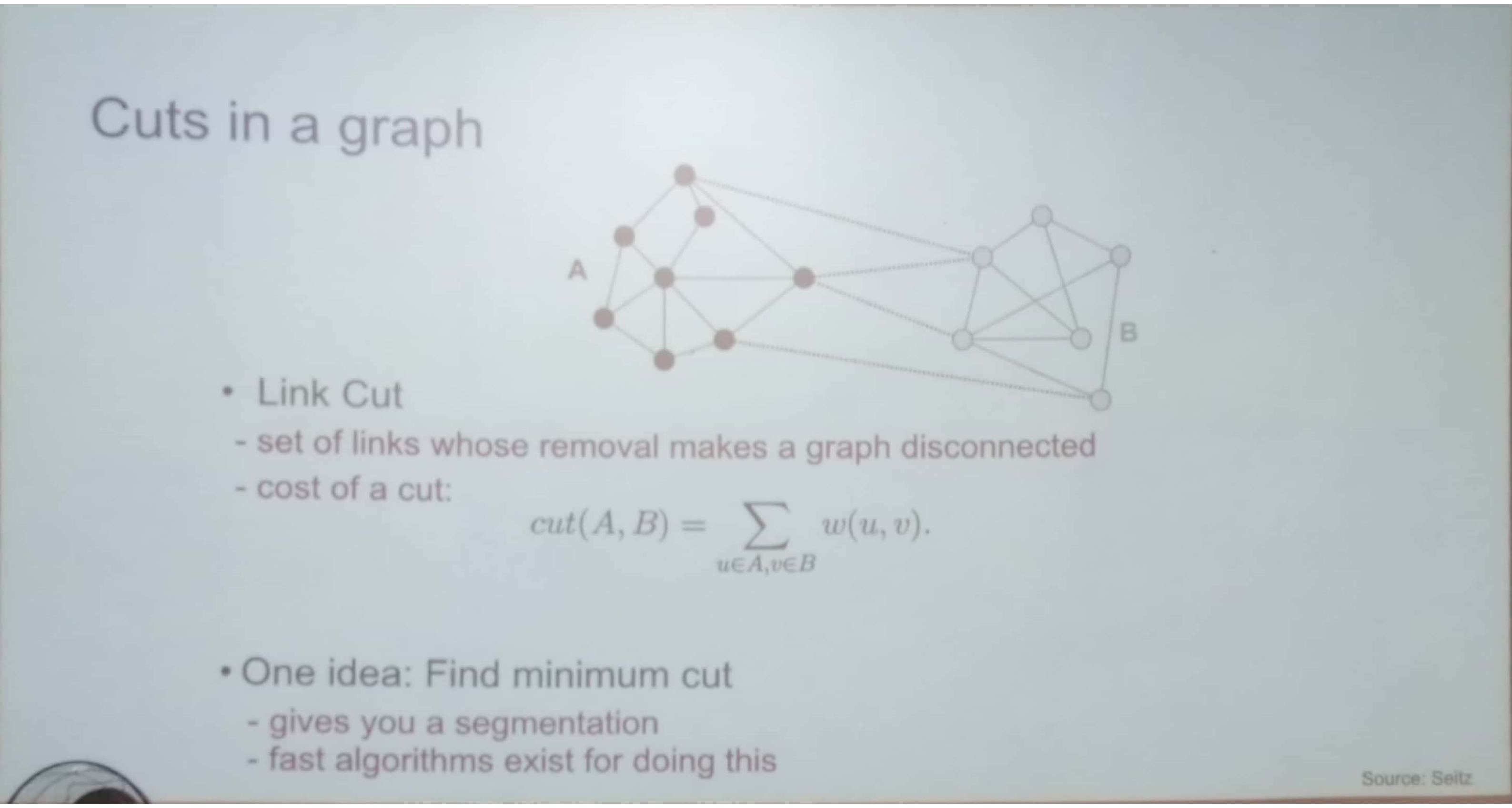
Segmentation as Optimal Labeling

- Model knowledge about the world.
- Classify each pixel as belonging to a specific object
  - Independent classification does not work
  - Need to incorporate neighborhood information
- Consider a graph over the image
  - Each node in the graph need to be labeled
  - Edges in the graph represent neighborhood constraints
- Define a cost function, $Q(f)$, using the above



Segmentation by Graph Cuts

- We Break Graph into Segments
  - –Delete links that cross between segments



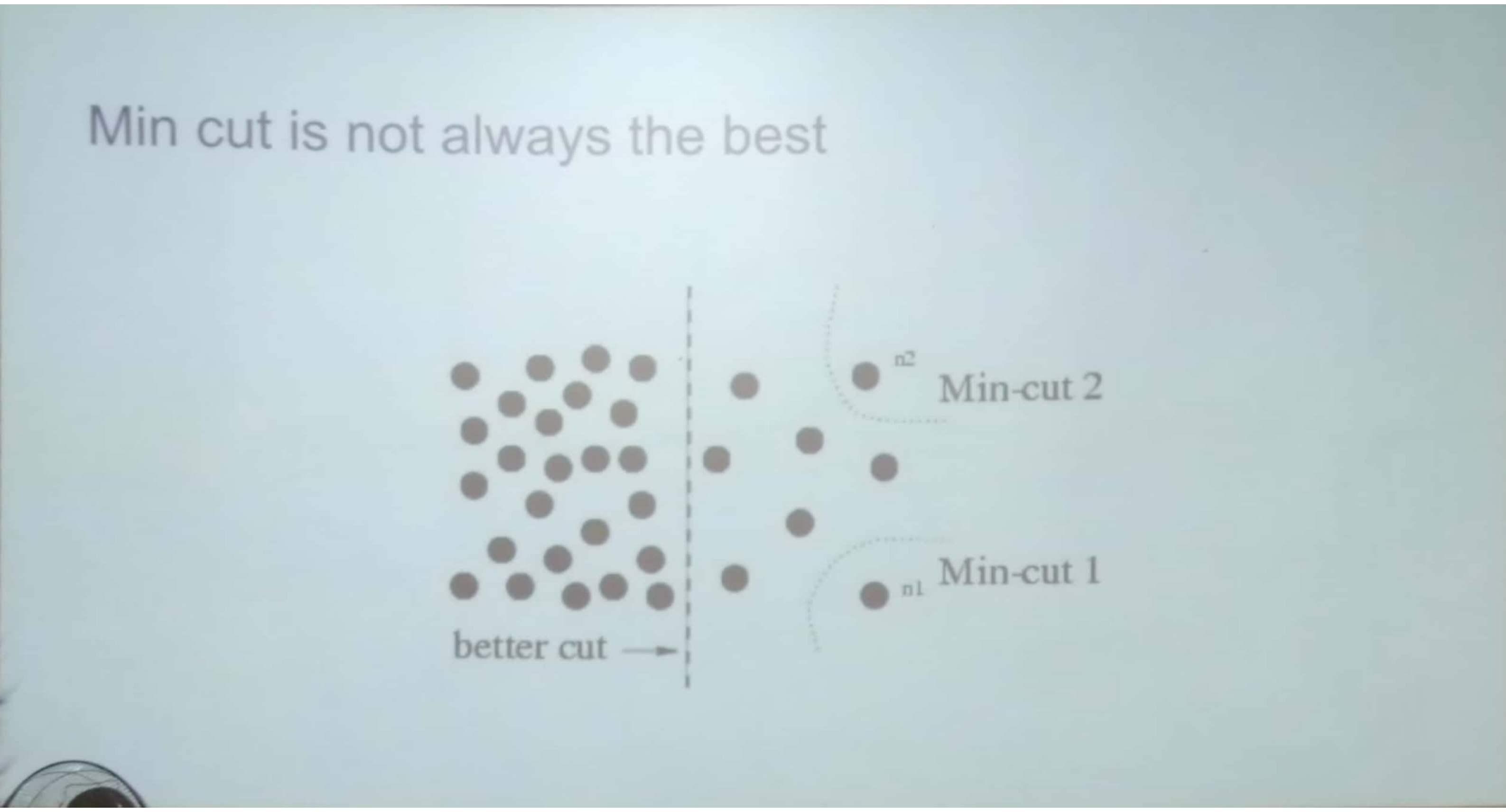
Cuts in a graph

- Link Cut
  - set of links whose removal makes a graph disconnected
  - cost of a cut:

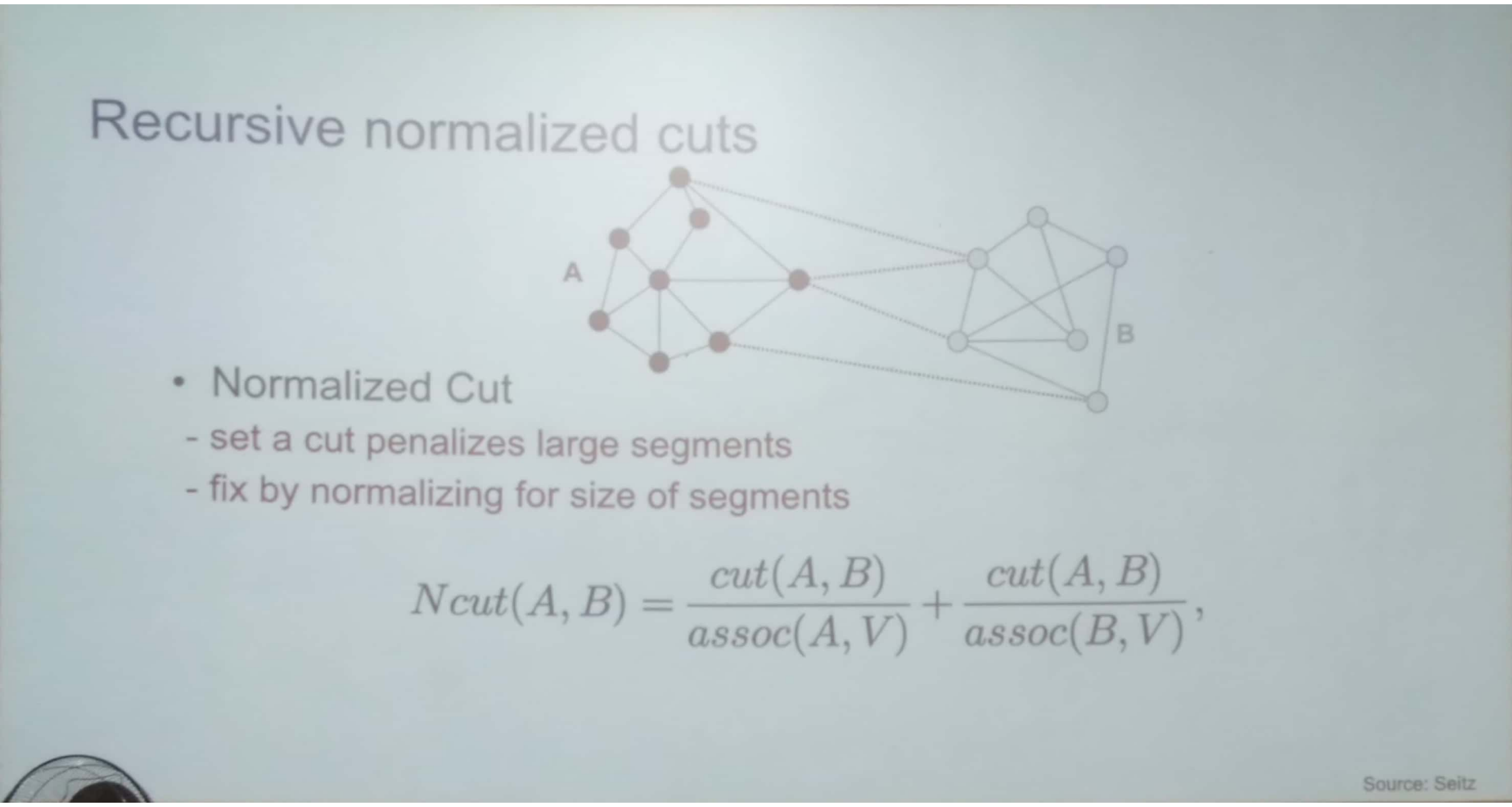
$$cut(A,B) = \sum_{u \in A, v \in B} w(u,v).$$

- One idea: Find minimum cut
  - gives you a segmentation
  - fast algorithms exist for doing this

Source: Seitz



Min cut is not always the best



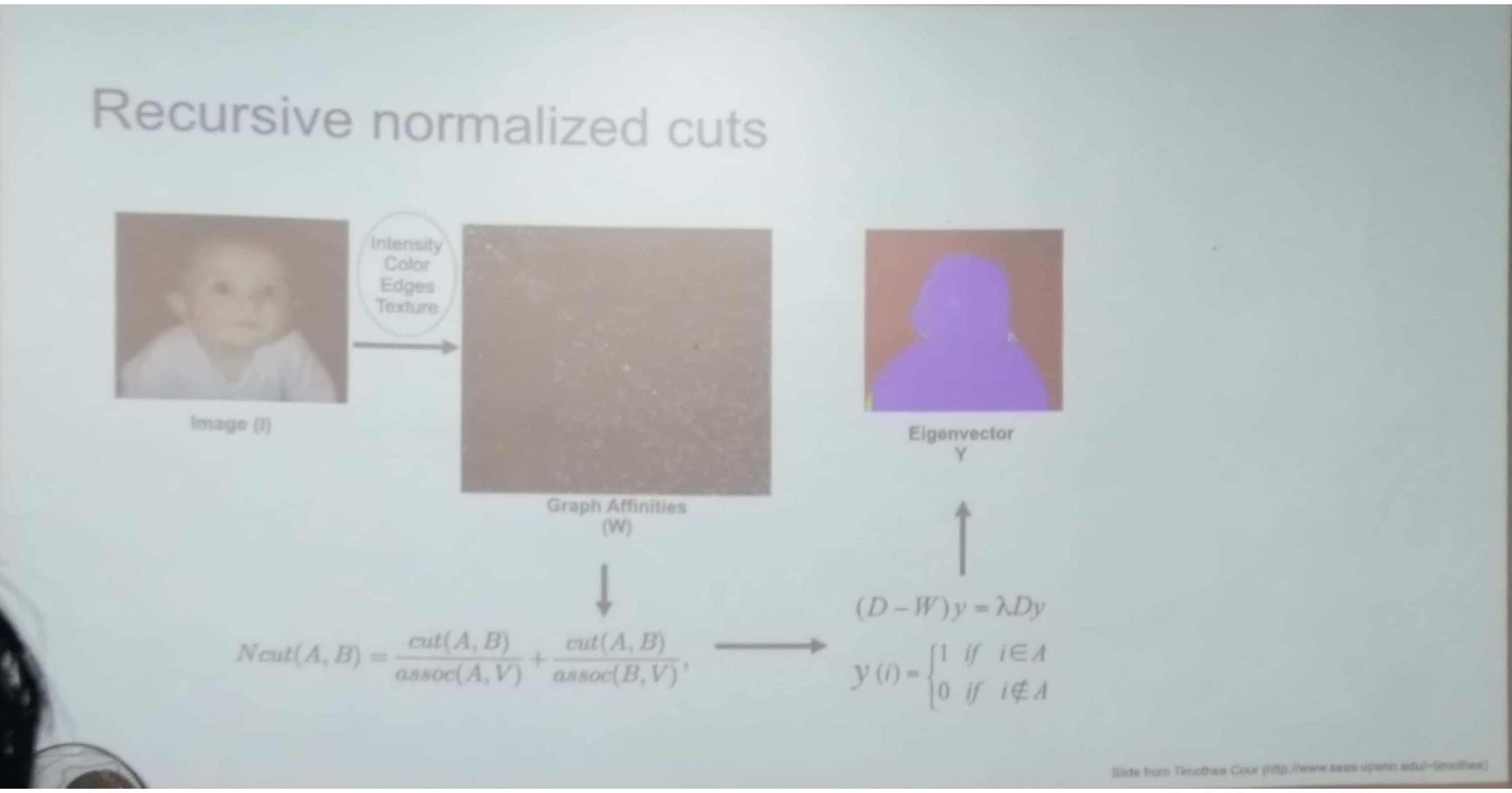
Recursive normalized cuts

- Normalized Cut

- set a cut penalizes large segments
- fix by normalizing for size of segments

$$Ncut(A,B) = \frac{cut(A,B)}{assoc(A,V)} + \frac{cut(A,B)}{assoc(B,V)},$$

Source: Seitz



Recursive normalized cuts

Intensity
Color
Edges
Texture

Image (I)

Graph Affinities
(W)

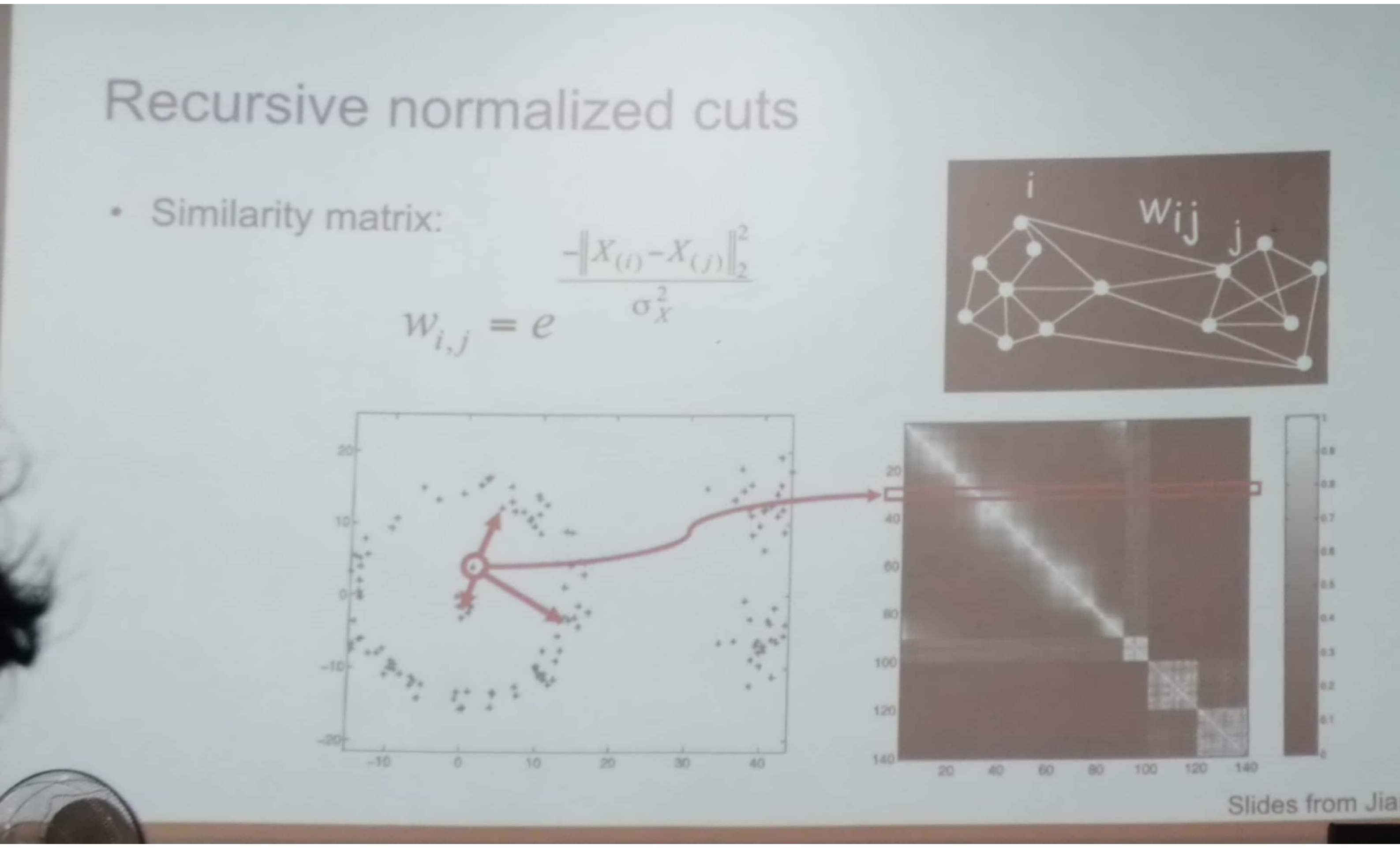
Eigenvector
Y

$$Ncut(A,B) = \frac{cut(A,B)}{assoc(A,V)} + \frac{cut(A,B)}{assoc(B,V)},$$

$$(D-W)y = \lambda D y$$

$$y(i) = \begin{cases} 1 & if \quad i \in A \\ 0 & if \quad i \notin A \end{cases}$$

Slide from Timothee Cour (http://www.seas.upenn.edu/~timothee)



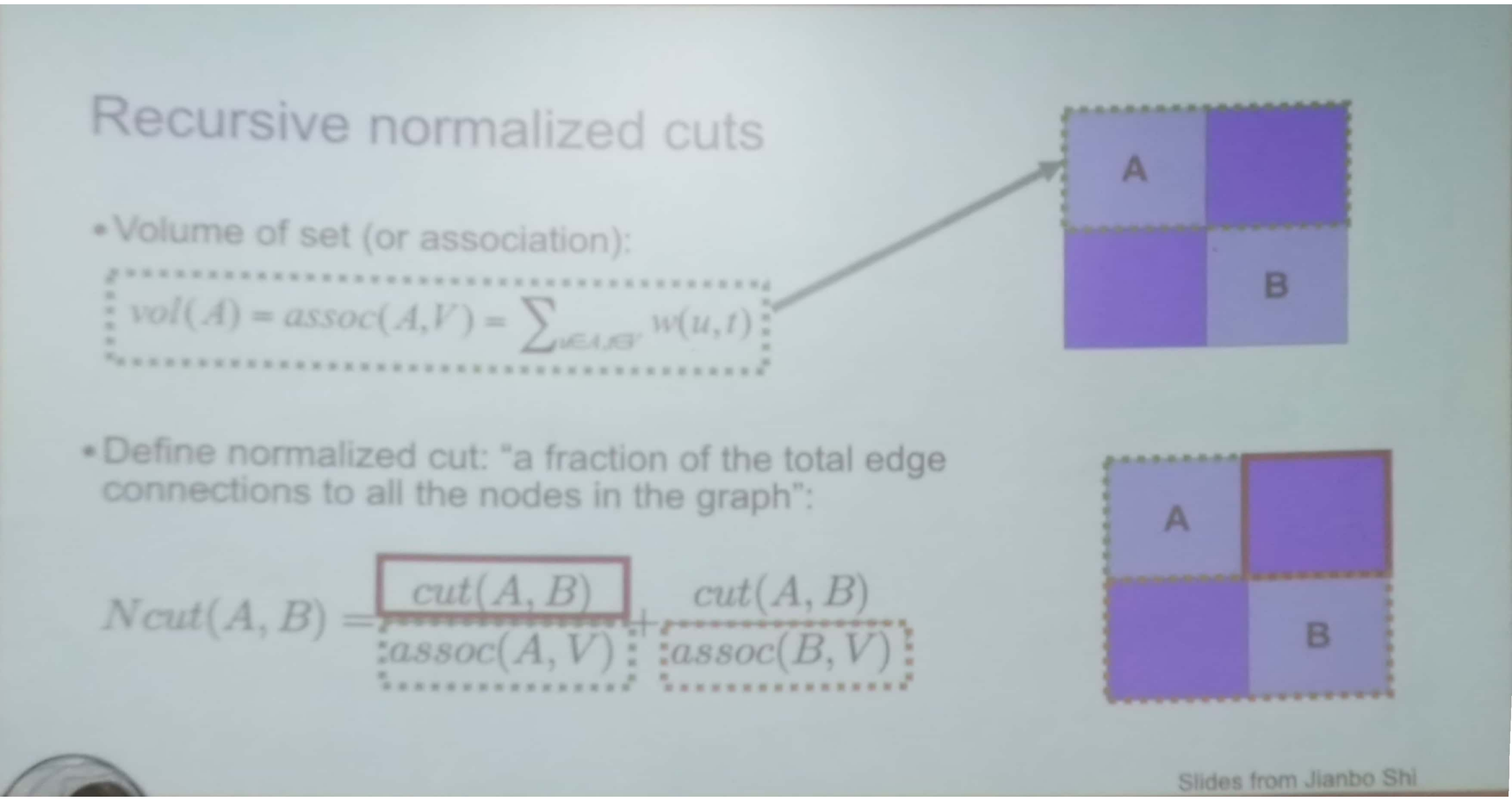
Recursive normalized cuts

- Similarity matrix:

$$w_{i,j} = e^{\frac{-\|X_{(i)}-X_{(j)}\|_2^2}{\sigma_X^2}}$$

i
wij
j
20
10
0
−10
−20
−10
0
10
20
30
40
20
40
60
80
100
120
140
20
40
60
80
100
120
140
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0

Slides from Jia



Recursive normalized cuts

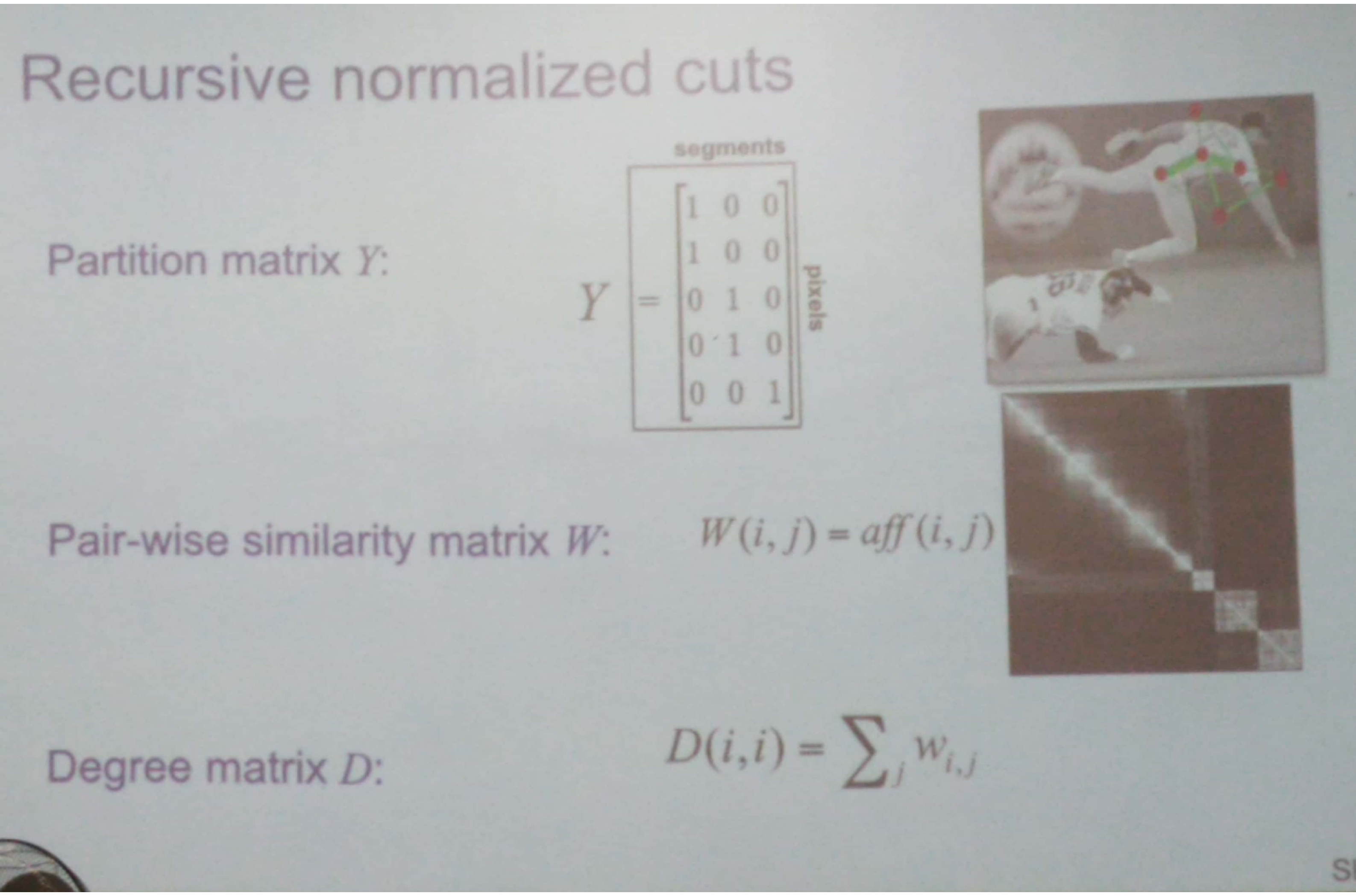
- Volume of set (or association):

$$vol(A) = assoc(A,V) = \sum_{u \in A, t \in V} w(u,t)$$

- Define normalized cut: "a fraction of the total edge connections to all the nodes in the graph":

$$Ncut(A,B) = \frac{\boxed{cut(A,B)}}{assoc(A,V)} + \frac{cut(A,B)}{assoc(B,V)}$$

Slides from Jianbo Shi



Recursive normalized cuts

Partition matrix Y:

$$Y = \begin{array}{cc} \text{segments} & \\ \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} & \text{pixels} \end{array}$$

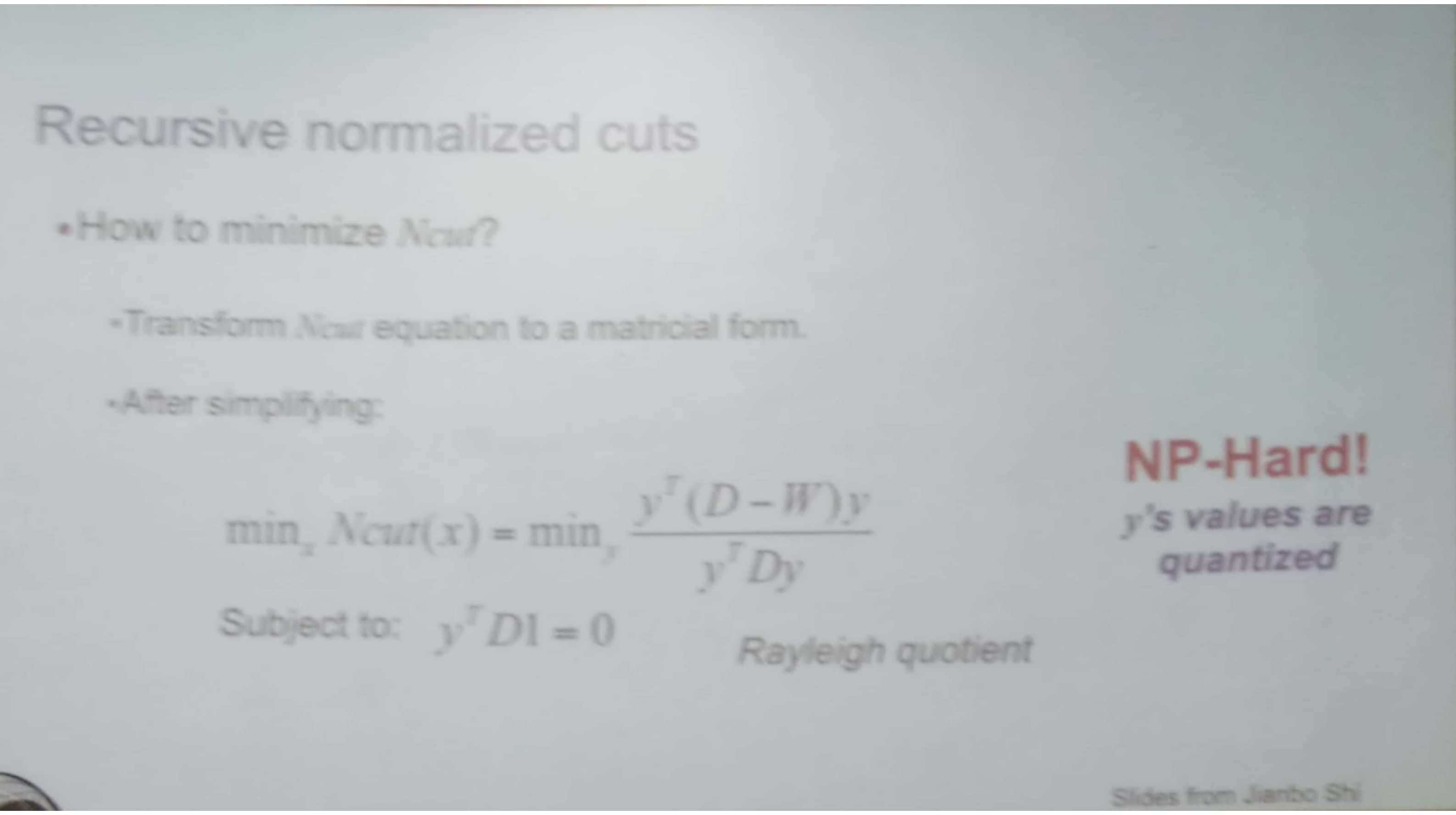
Pair-wise similarity matrix W:

$$W(i,j) = aff(i,j)$$

Degree matrix D:

$$D(i,i) = \sum_j w_{i,j}$$

Sl



Recursive normalized cuts

- How to minimize *Ncut*?
  - Transform *Ncut* equation to a matricial form.
  - After simplifying:

$$\min_x Ncut(x) = \min_y \frac{y^T(D-W)y}{y^T D y}$$

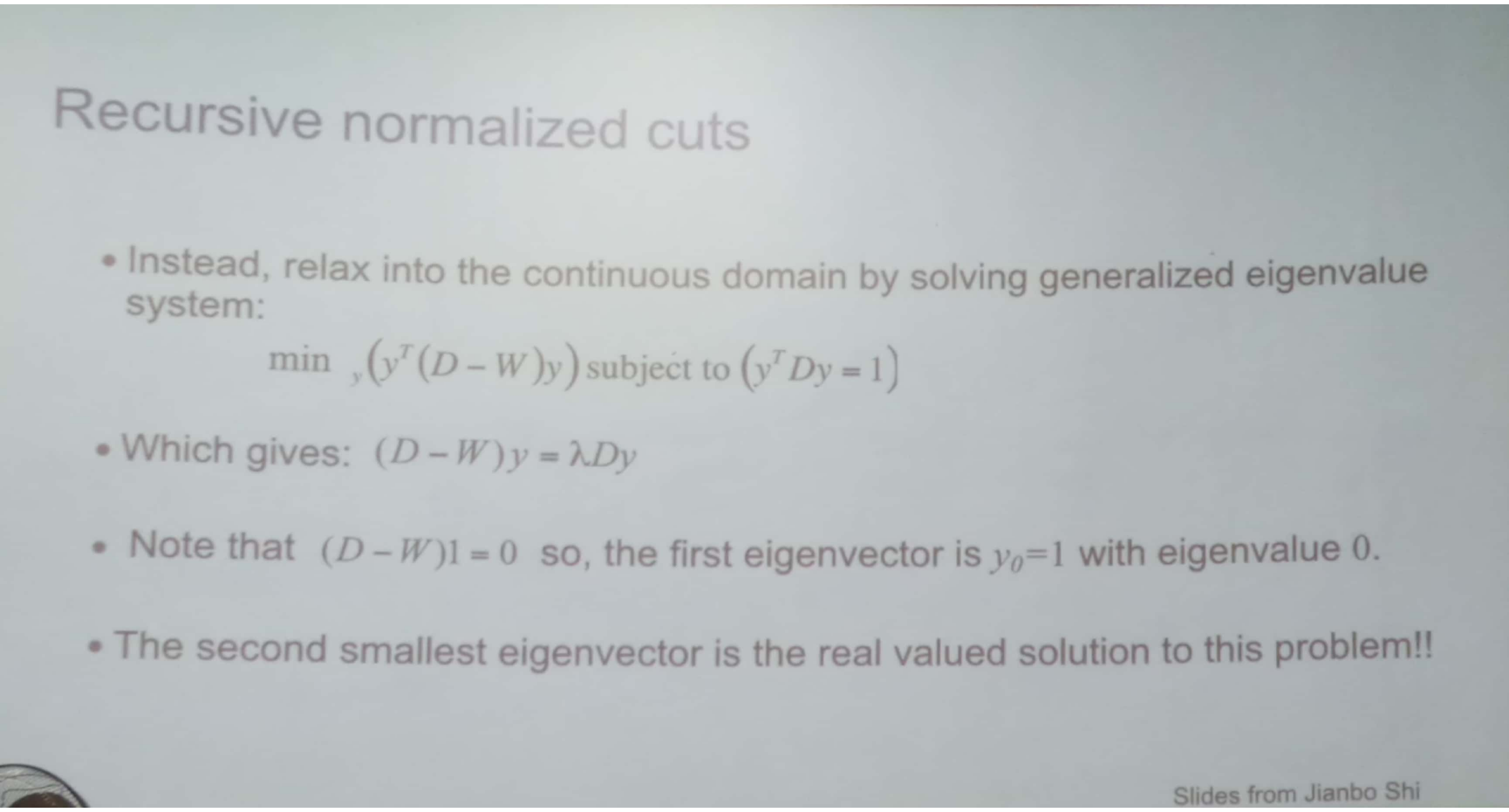
Subject to: $y^T D1 = 0$

Rayleigh quotient

NP-Hard!

y's values are quantized

Slides from Jianbo Shi



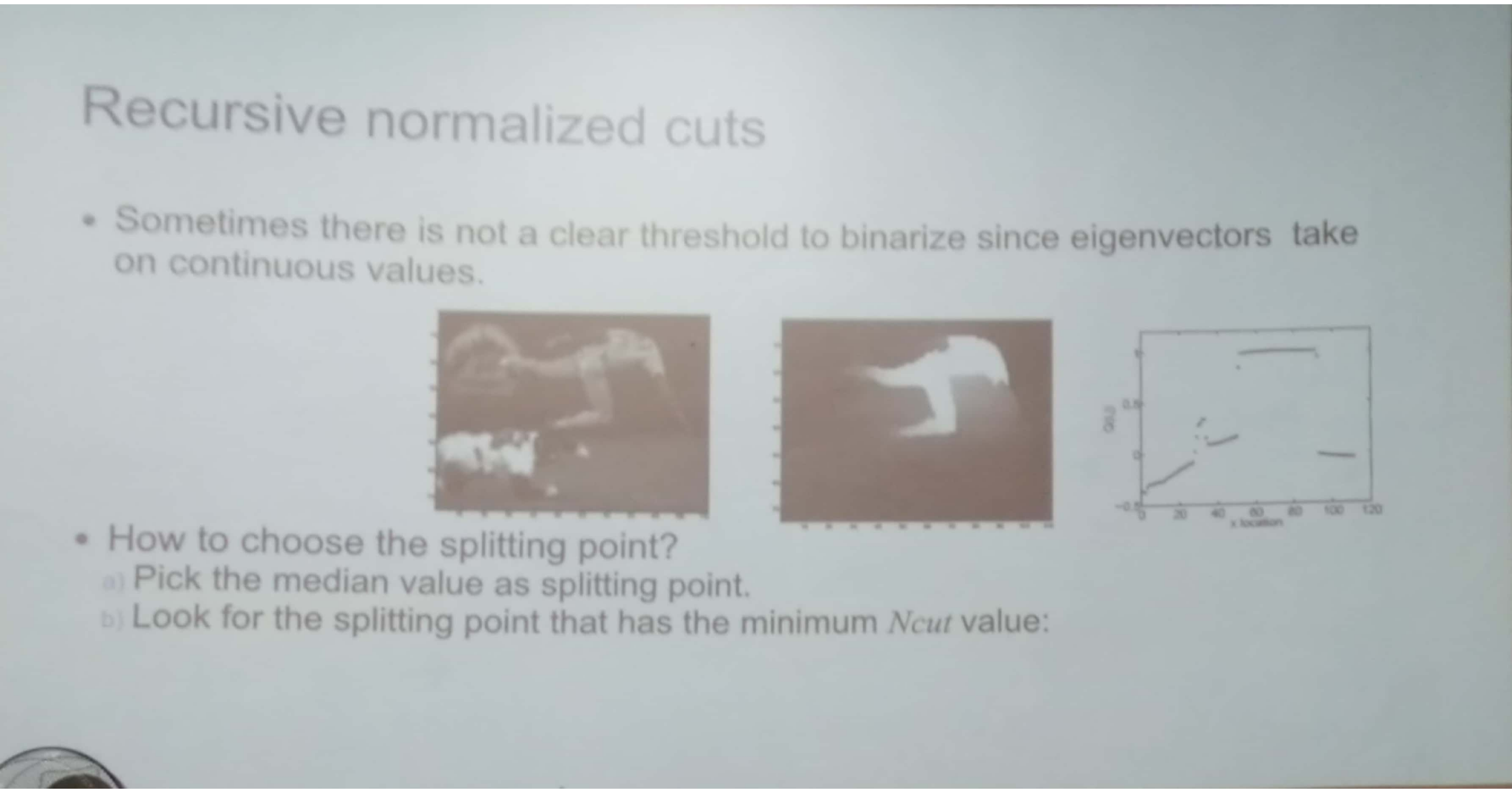
Recursive normalized cuts

- Instead, relax into the continuous domain by solving generalized eigenvalue system:

$$\min_y \left(y^T(D-W)y\right) \text{ subject to } \left(y^T D y = 1\right)$$

- Which gives: $(D-W)y = \lambda D y$
- Note that $(D-W)1 = 0$ so, the first eigenvector is $y_0=1$ with eigenvalue 0.
- The second smallest eigenvector is the real valued solution to this problem!!

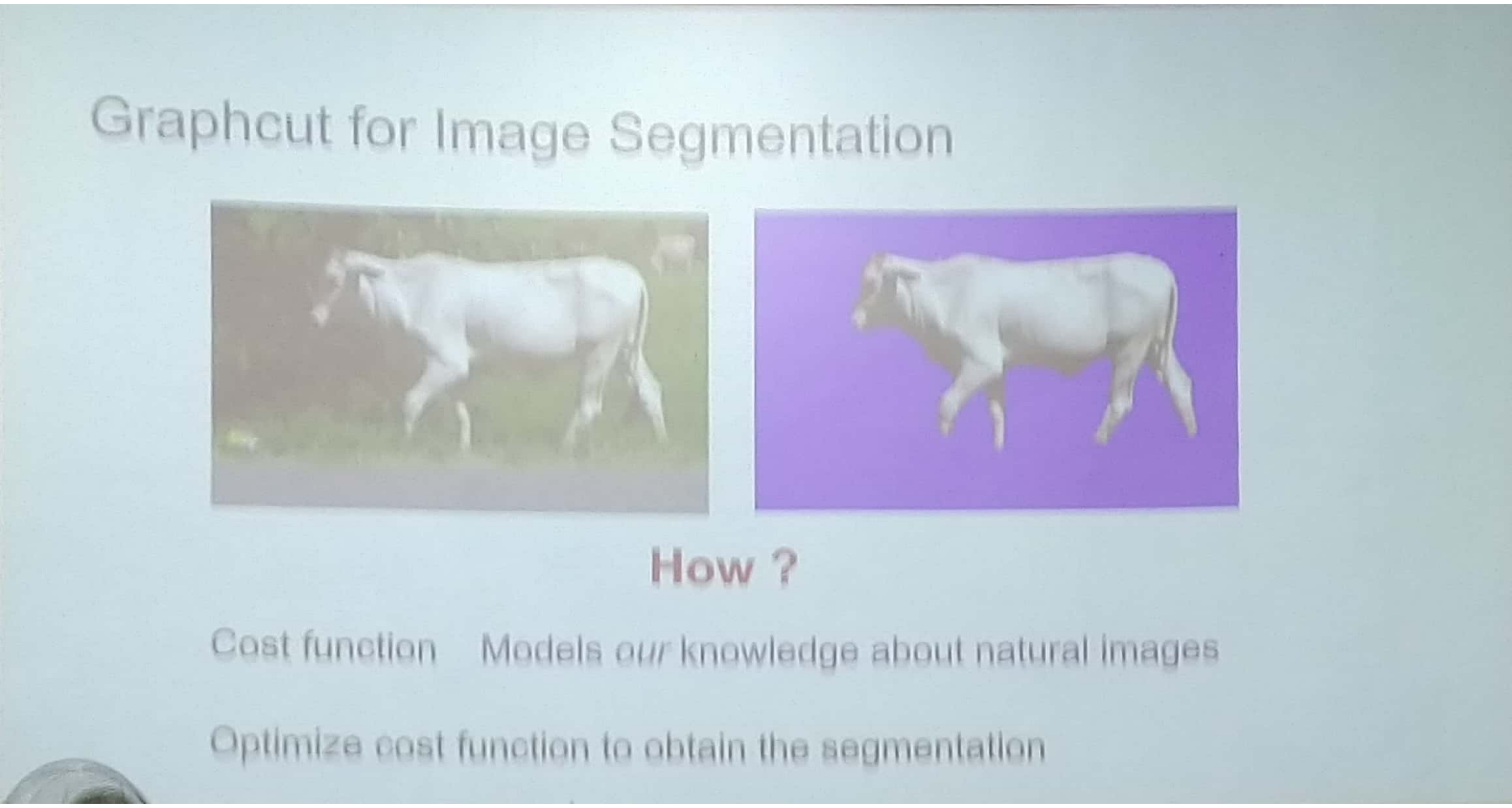
Slides from Jianbo Shi



Recursive normalized cuts

- Sometimes there is not a clear threshold to binarize since eigenvectors take on continuous values.

- How to choose the splitting point?
  a) Pick the median value as splitting point.
  b) Look for the splitting point that has the minimum *Ncut* value:

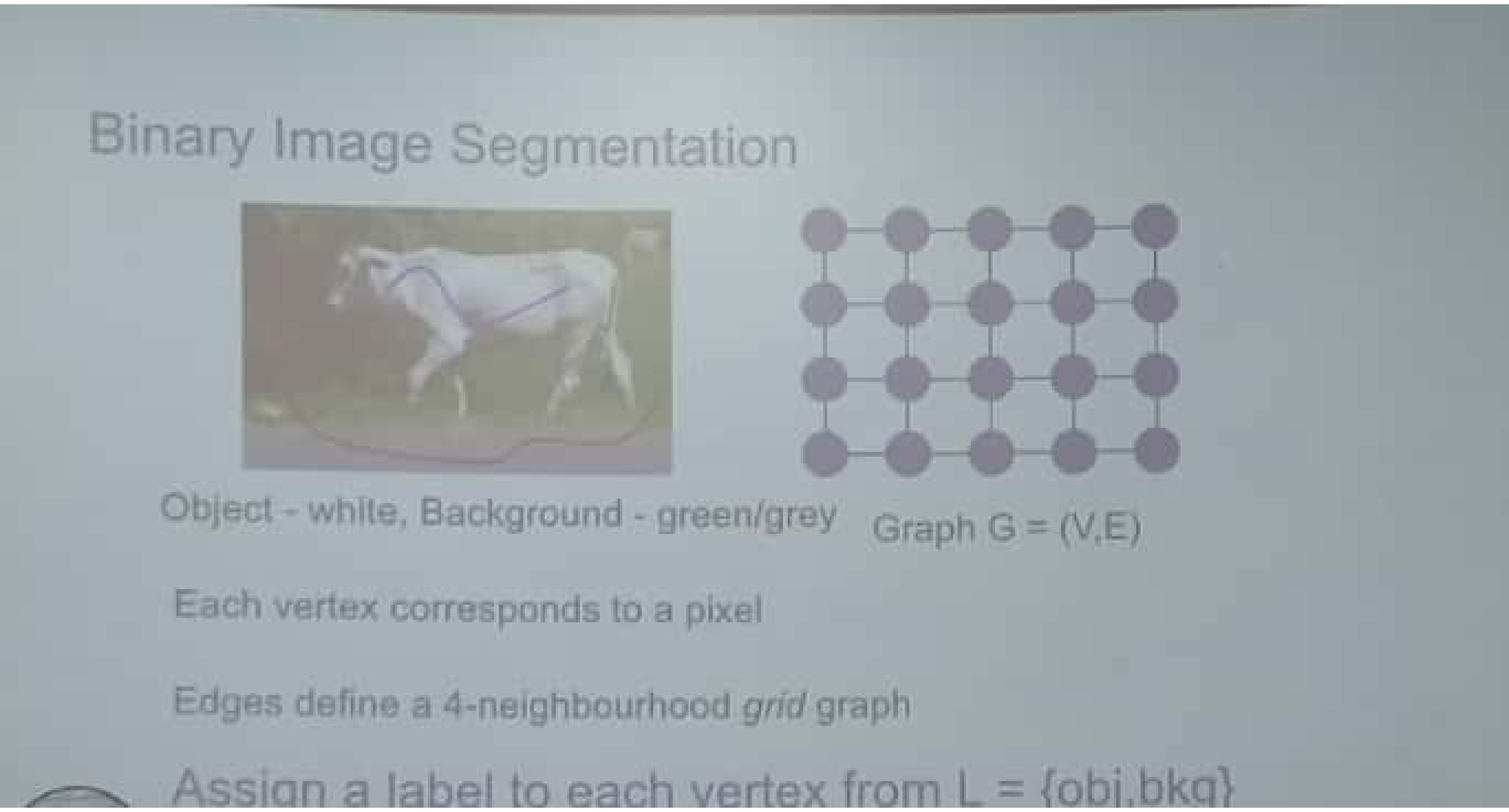


Graphcut for Image Segmentation

How ?

Cost function   Models *our* knowledge about natural images

Optimize cost function to obtain the segmentation



Binary Image Segmentation

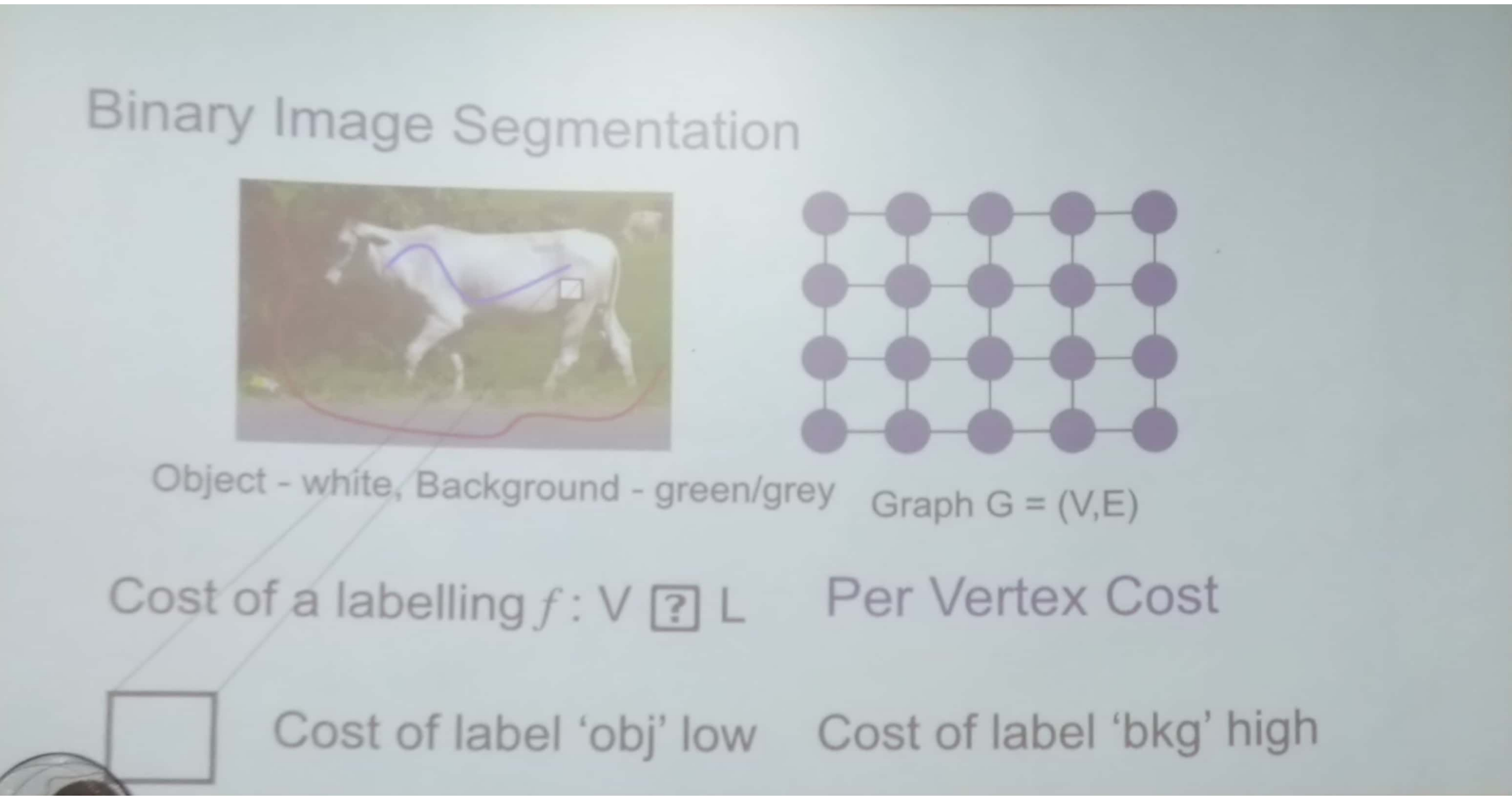
Object - white, Background - green/grey

Graph G = (V,E)

Each vertex corresponds to a pixel

Edges define a 4-neighbourhood *grid* graph

Assign a label to each vertex from L = {obj,bkg}



Binary Image Segmentation

Object - white, Background - green/grey

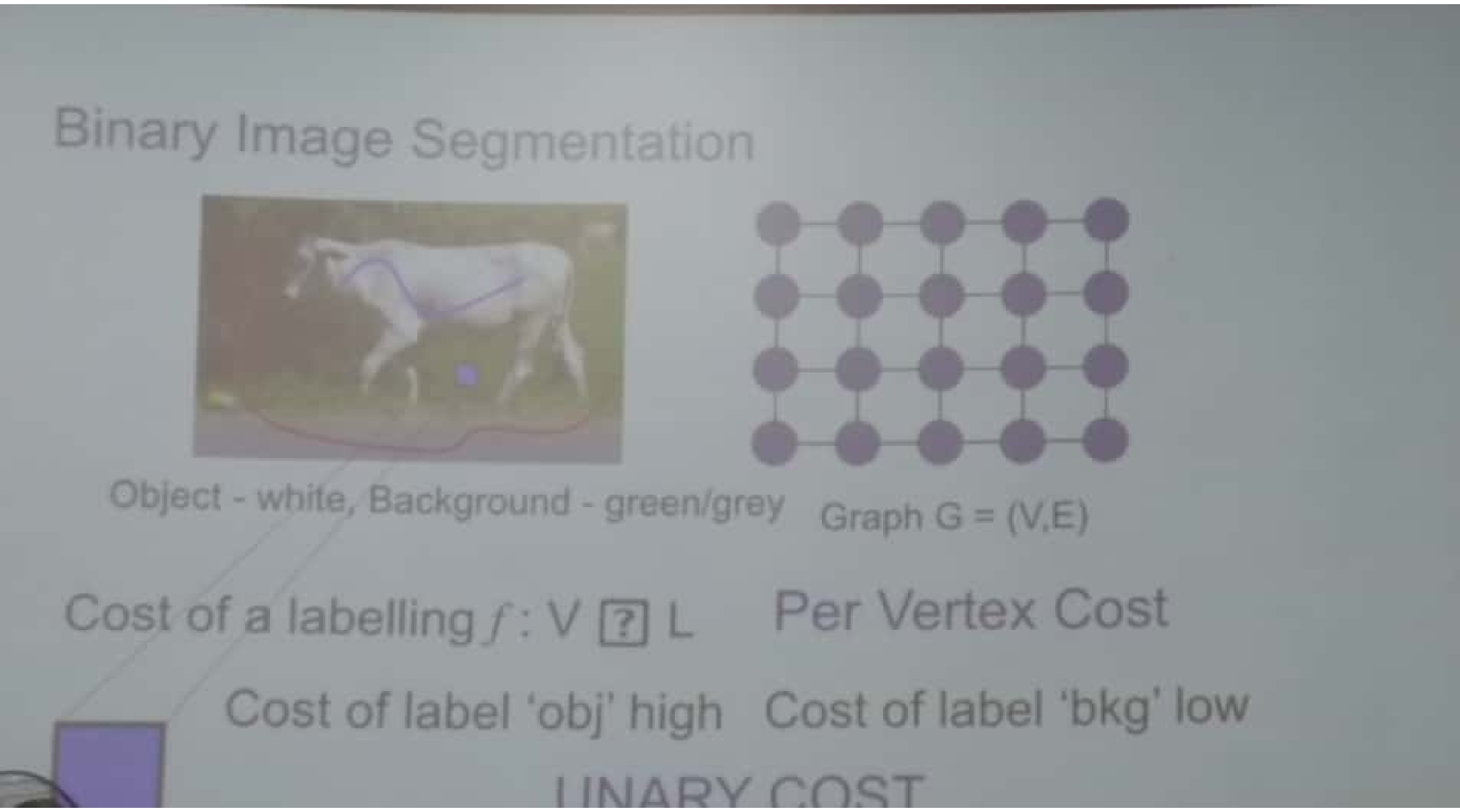
Graph G = (V,E)

Cost of a labelling f : V $\boxed{?}$ L

Per Vertex Cost

□ Cost of label ‘obj’ low

Cost of label ‘bkg’ high



Binary Image Segmentation

Object - white, Background - green/grey

Graph G = (V,E)

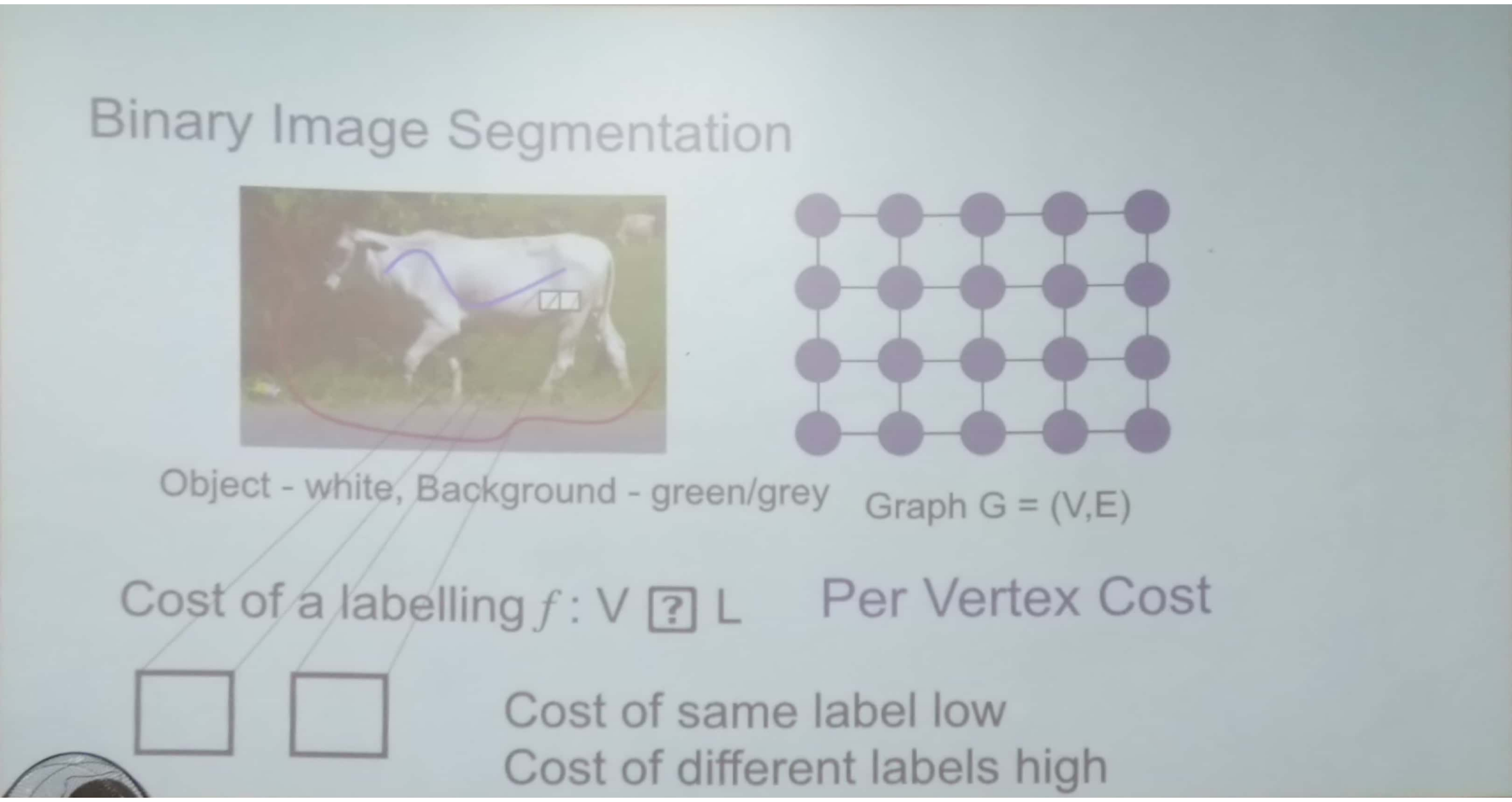
Cost of a labelling f : V ⍰ L

Per Vertex Cost

Cost of label 'obj' high

Cost of label 'bkg' low

UNARY COST



Binary Image Segmentation

Object - white, Background - green/grey

Graph G = (V,E)

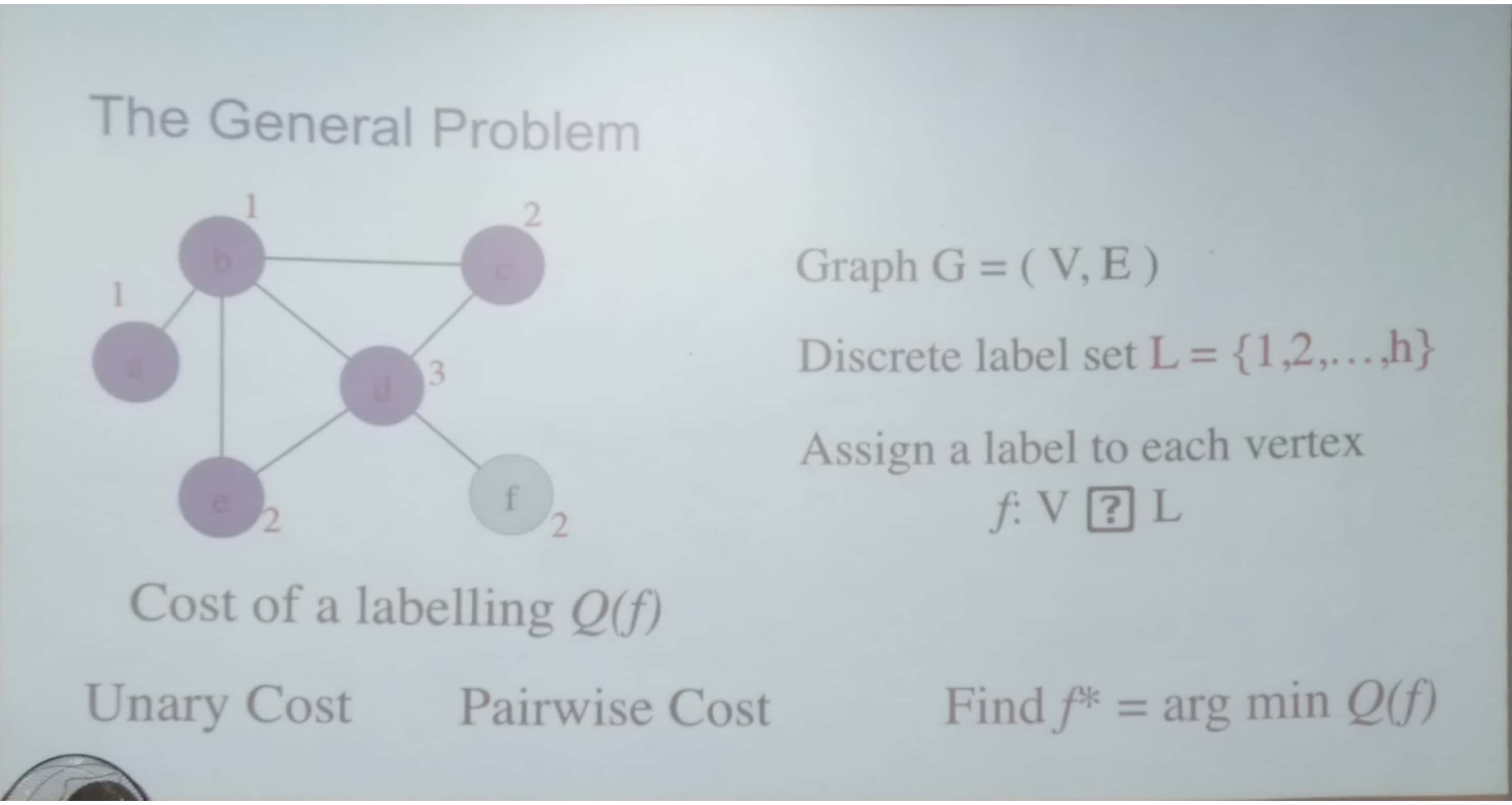
Cost of a labelling f : V $\boxed{?}$ L

Per Vertex Cost

□ □

Cost of same label low

Cost of different labels high


b
1
c
2
a
1
d
3
e
2
f
2

The General Problem

Graph G = (V, E)

Discrete label set L = {1,2,…,h}

Assign a label to each vertex

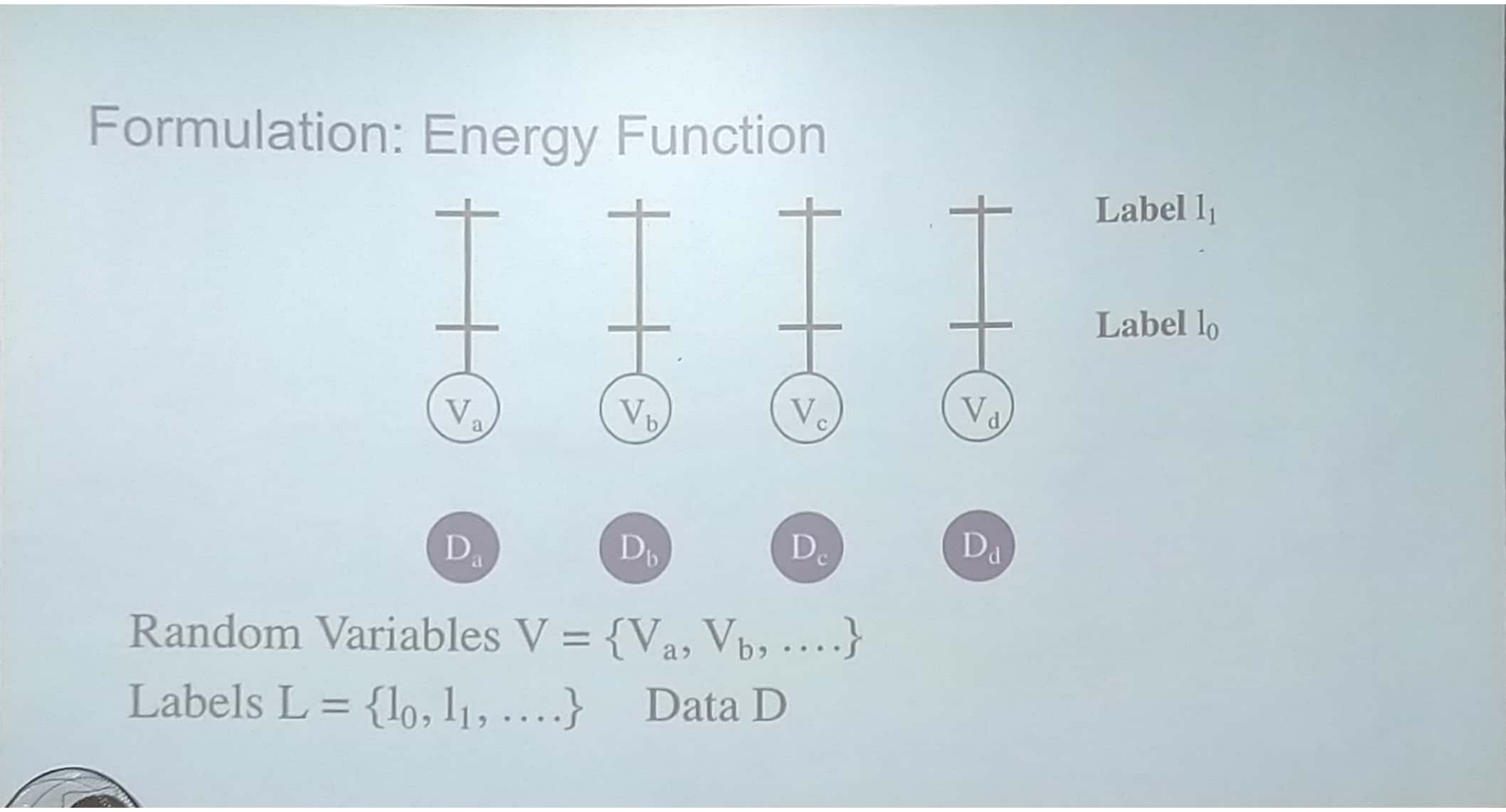
f: V $\boxed{?}$ L

Cost of a labelling $Q(f)$

Unary Cost

Pairwise Cost

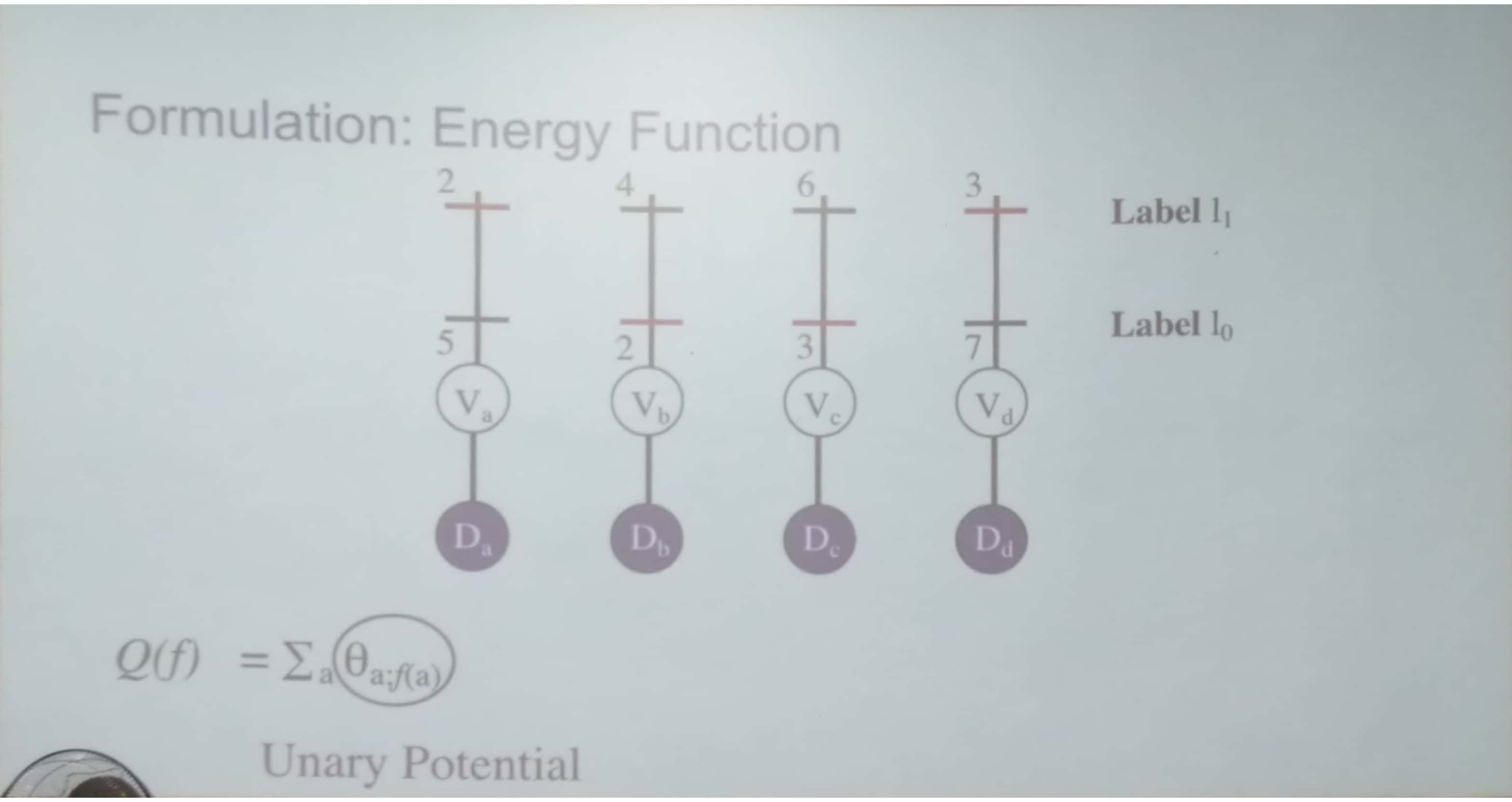
Find f^* = arg min $Q(f)$



Formulation: Energy Function

Random Variables $V = \{V_a, V_b, \ldots.\}$

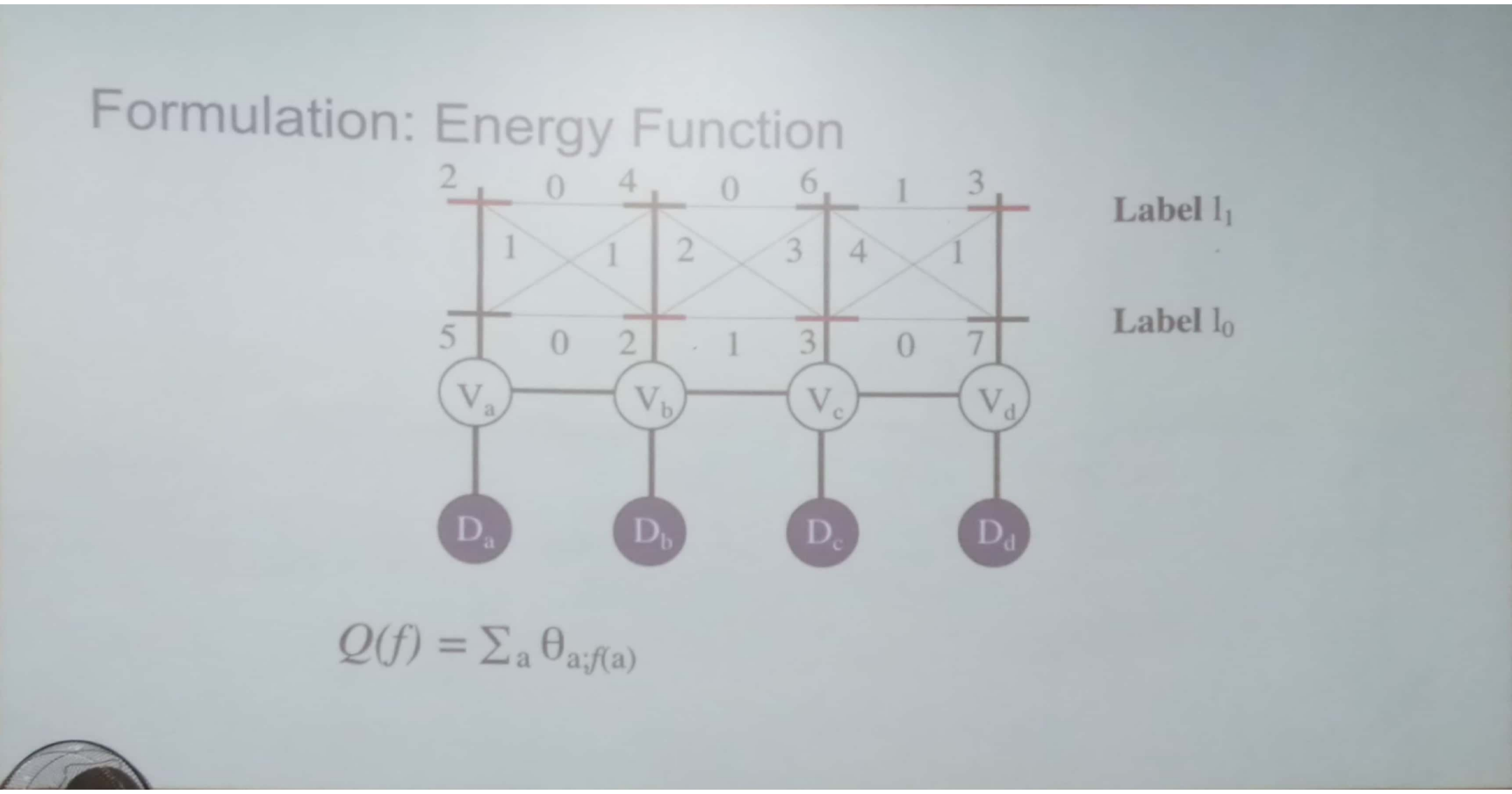
Labels $L = \{l_0, l_1, \ldots.\}$ Data D



Formulation: Energy Function

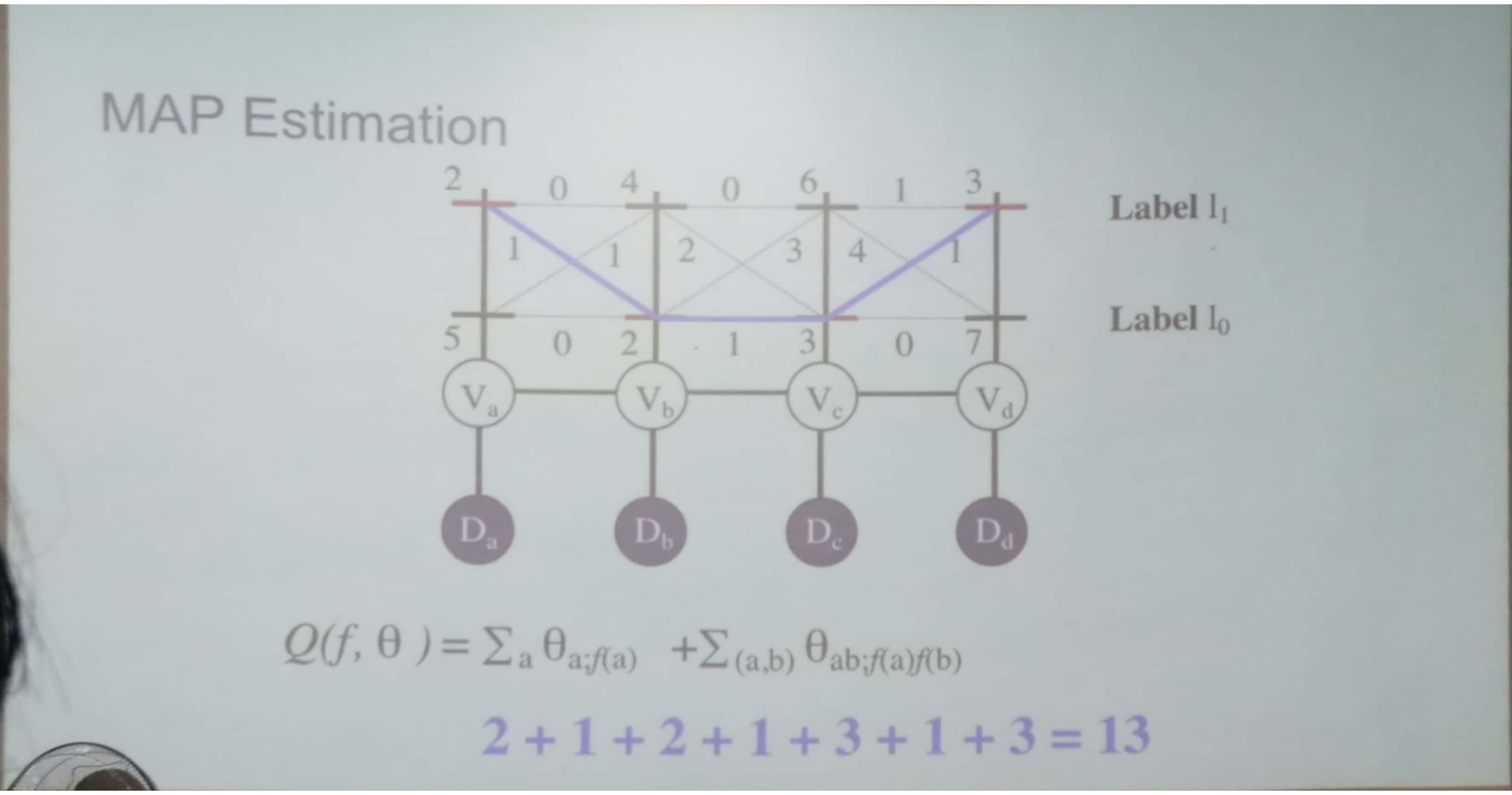
$$Q(f) = \sum_a \boxed{\theta_{a;f(a)}}$$

Unary Potential



Formulation: Energy Function

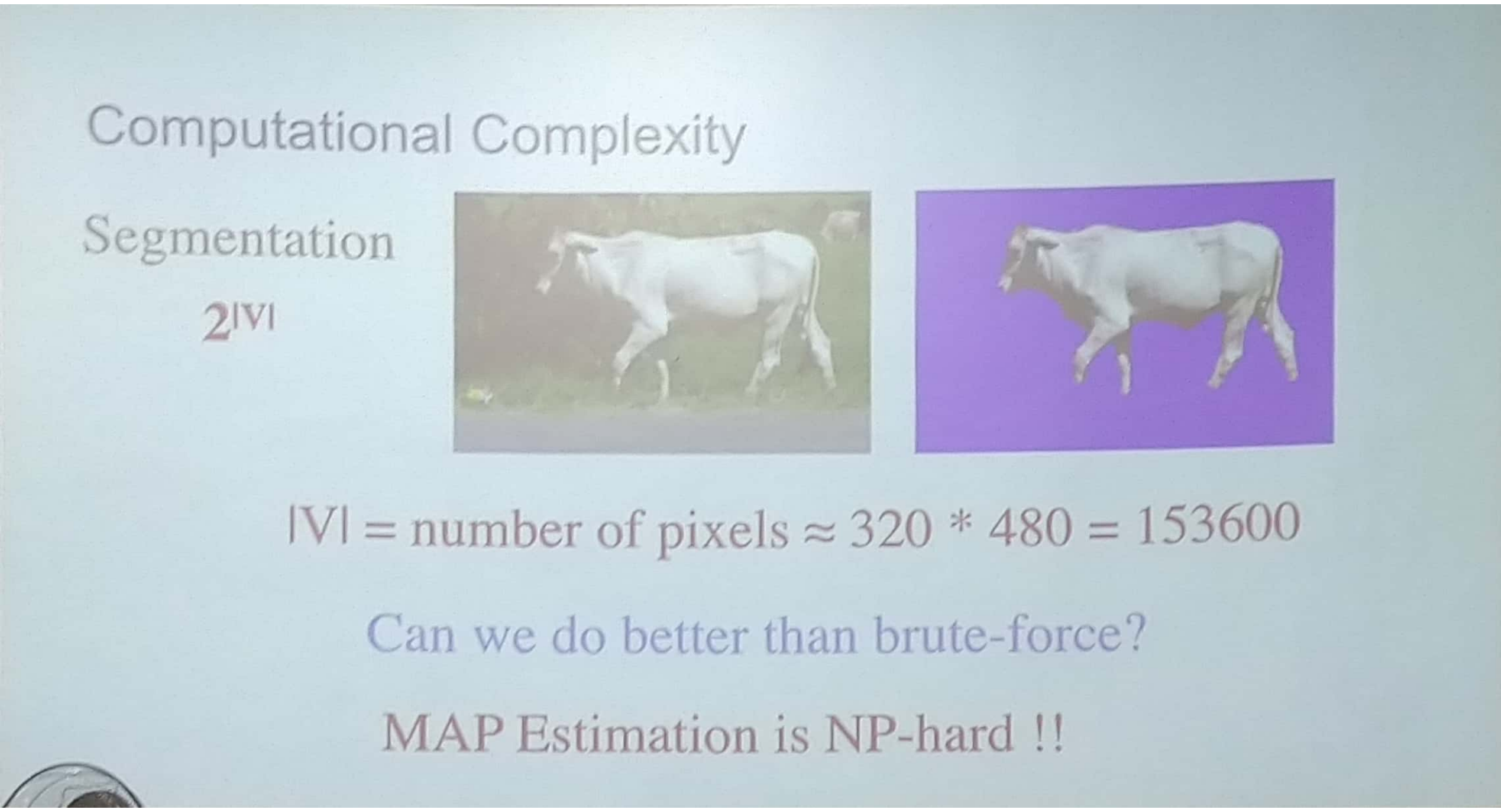
$$Q(f) = \Sigma_a \, \theta_{a;f(a)}$$



MAP Estimation

$$Q(f, \theta) = \sum_a \theta_{a;f(a)} \quad + \sum_{(a,b)} \theta_{ab;f(a)f(b)}$$

2 + 1 + 2 + 1 + 3 + 1 + 3 = 13



Computational Complexity

Segmentation

$2^{|V|}$

$|V|$ = number of pixels ≈ 320 * 480 = 153600

Can we do better than brute-force?

MAP Estimation is NP-hard !!