

Lecture 04 - Component Segmentation

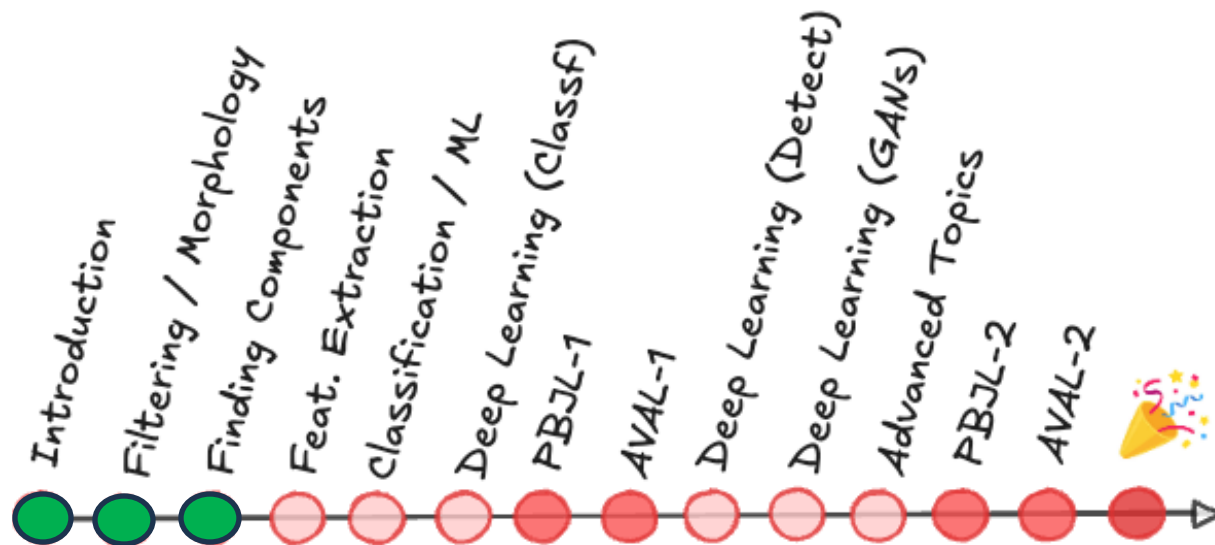
Prof. André Gustavo Hochuli

gustavo.hochuli@pucpr.br

aghochuli@ppgia.pucpr.br

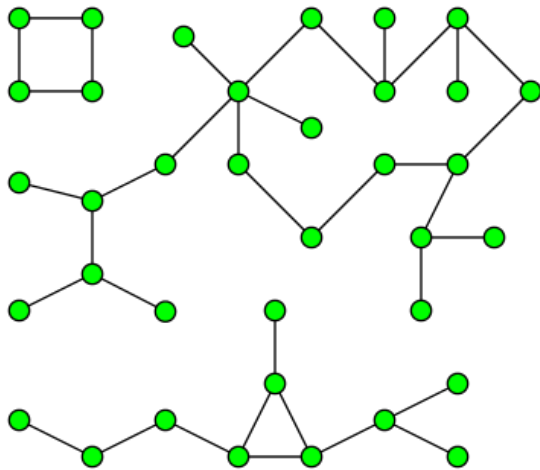
Topics

- Discussion of Practice 03
- Component Segmentation
 - Finding Connected Components
 - Filtering Components
- Practice
 - License Plate Characters Segmentation



Component Segmentation

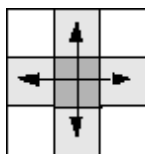
- A.K.A Connected Component Extraction, Blob Extraction,
- Its application comes from Graph Theory
 - Social Networks
 - Biology
 - Pattern Recognition



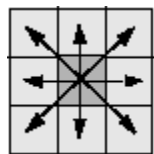
Connected Component Labelling

- Analyzes the non-zero pixel's neighborhood (foreground)
- Label each connected pixel with a label (1,2,3,4....)

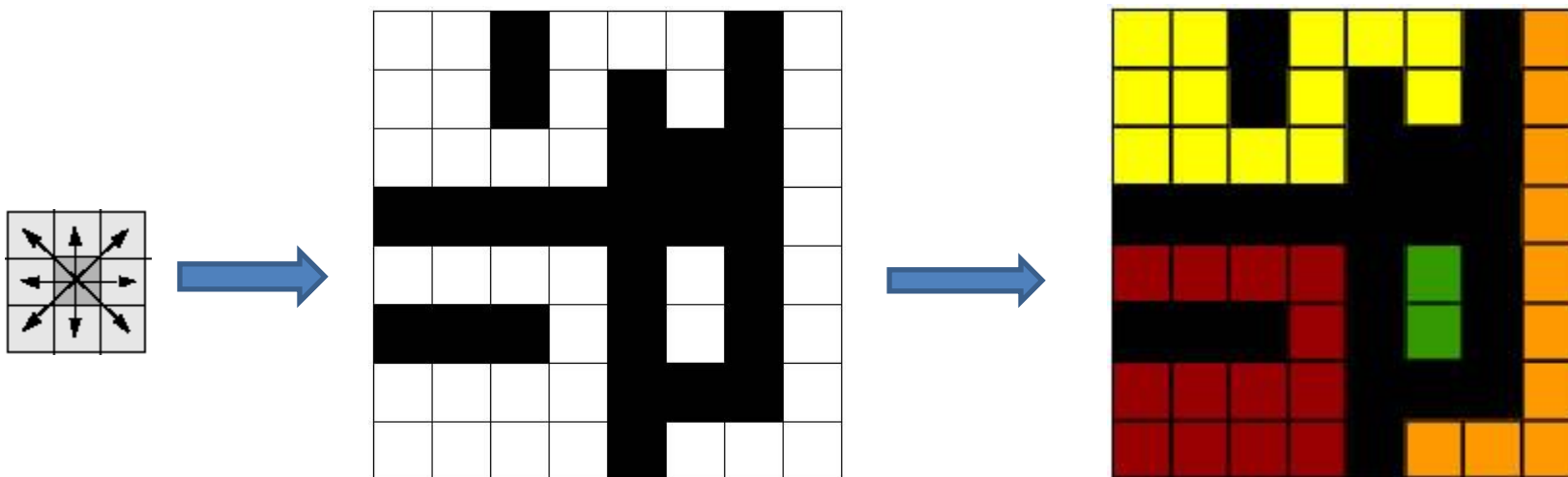
- Kernels:



4-Neighbors



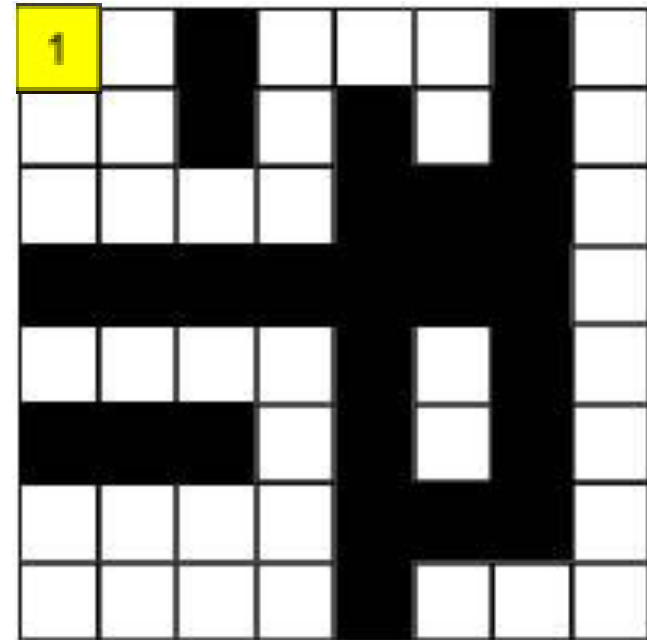
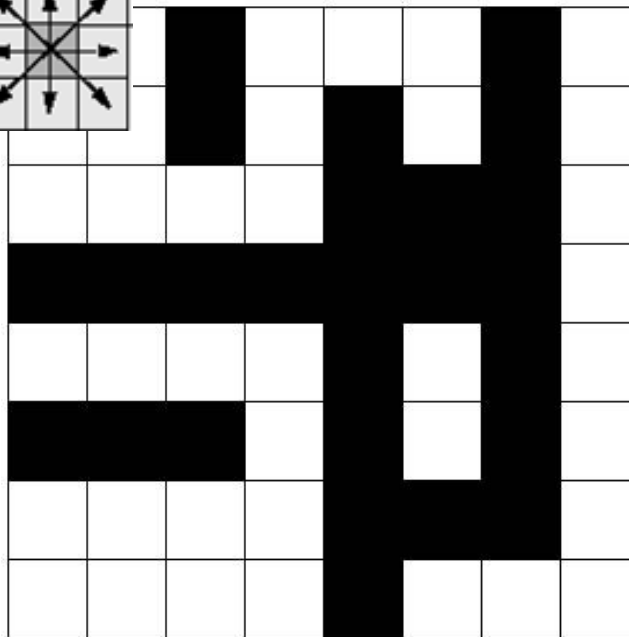
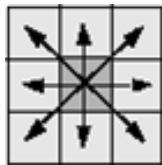
8-Neighbors



Row by Row Algorithm

- Sliding a connectivity kernel , row by row (2 passes)
 - If the center falls in a non-zero pixel, label it!
 - Labeling:
 - If there are no labeled pixels connected, attribute a new label
 - Otherwise, attribute to it the neighbor's label.
 - A Union-Find structure control adjacent labels (Union-Find)

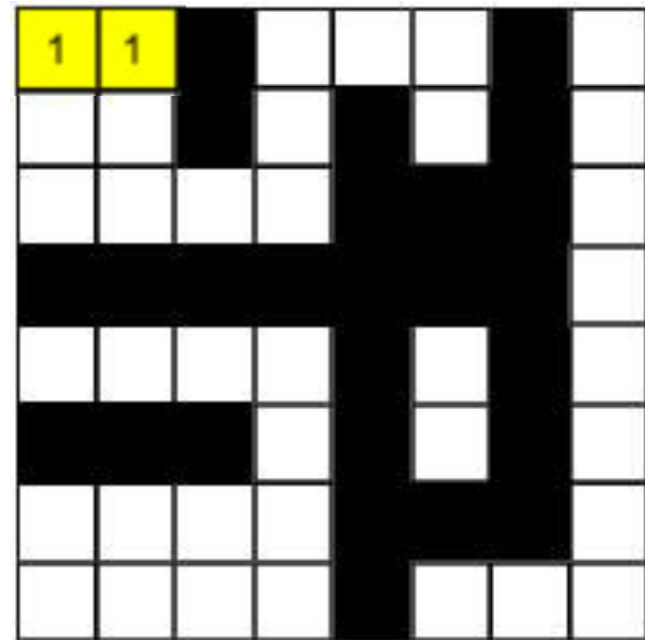
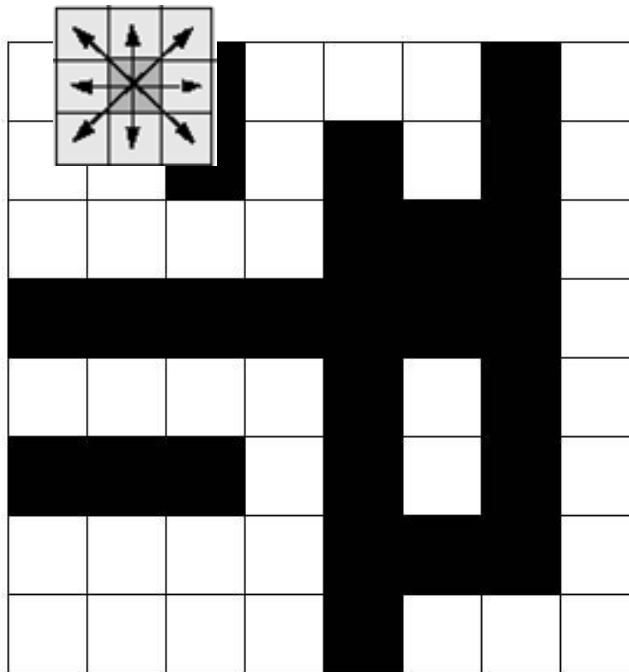
- **Pass #1:**
 - **Row #1**



Row by Row Algorithm

- Sliding a connectivity kernel , row by row (2 passes)
 - If the center falls in a non-zero pixel, label it!
 - Labeling:
 - If there are no labeled pixels connected, attribute a new label
 - Otherwise, attribute to it the neighbor 's label.
 - A Union-Find structure control adjacent labels (Union-Find)

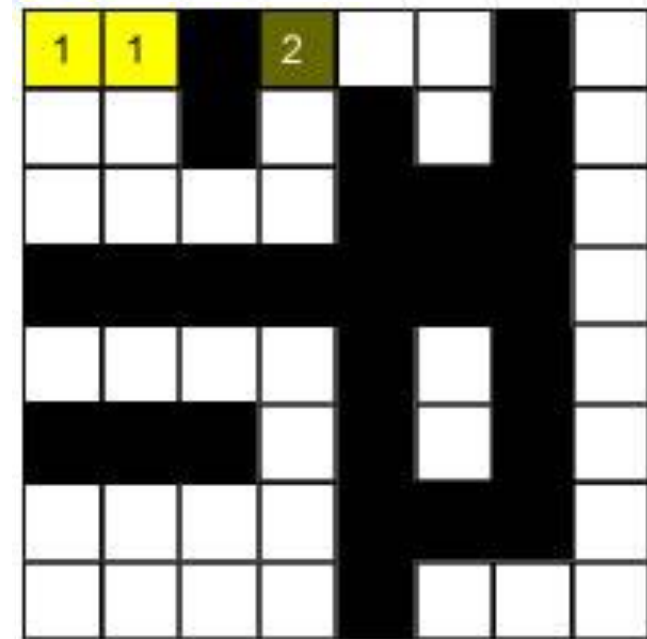
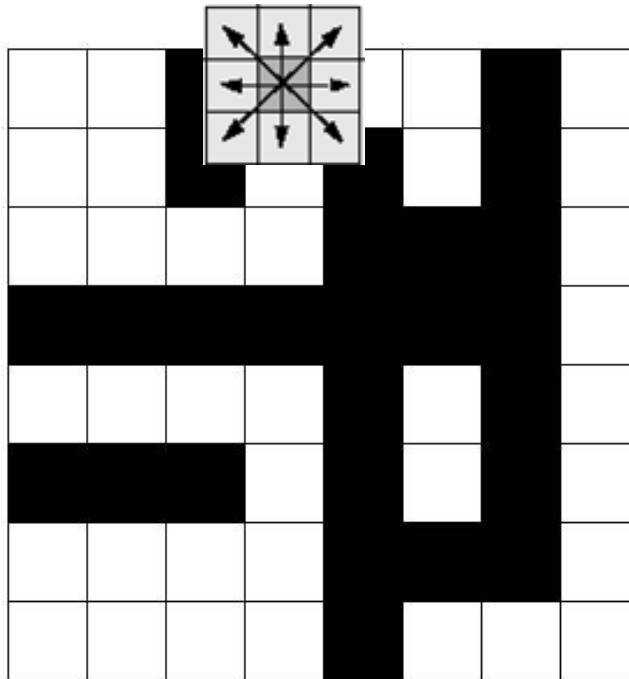
- **Pass #1:**
 - **Row #1**



Row by Row Algorithm

- Sliding a connectivity kernel , row by row (2 passes)
 - If the center falls in a non-zero pixel, label it!
 - Labeling:
 - If there are no labeled pixels connected, attribute a new label
 - Otherwise, attribute to it the neighbor 's label.
 - A Union-Find structure control adjacent labels (Union-Find)

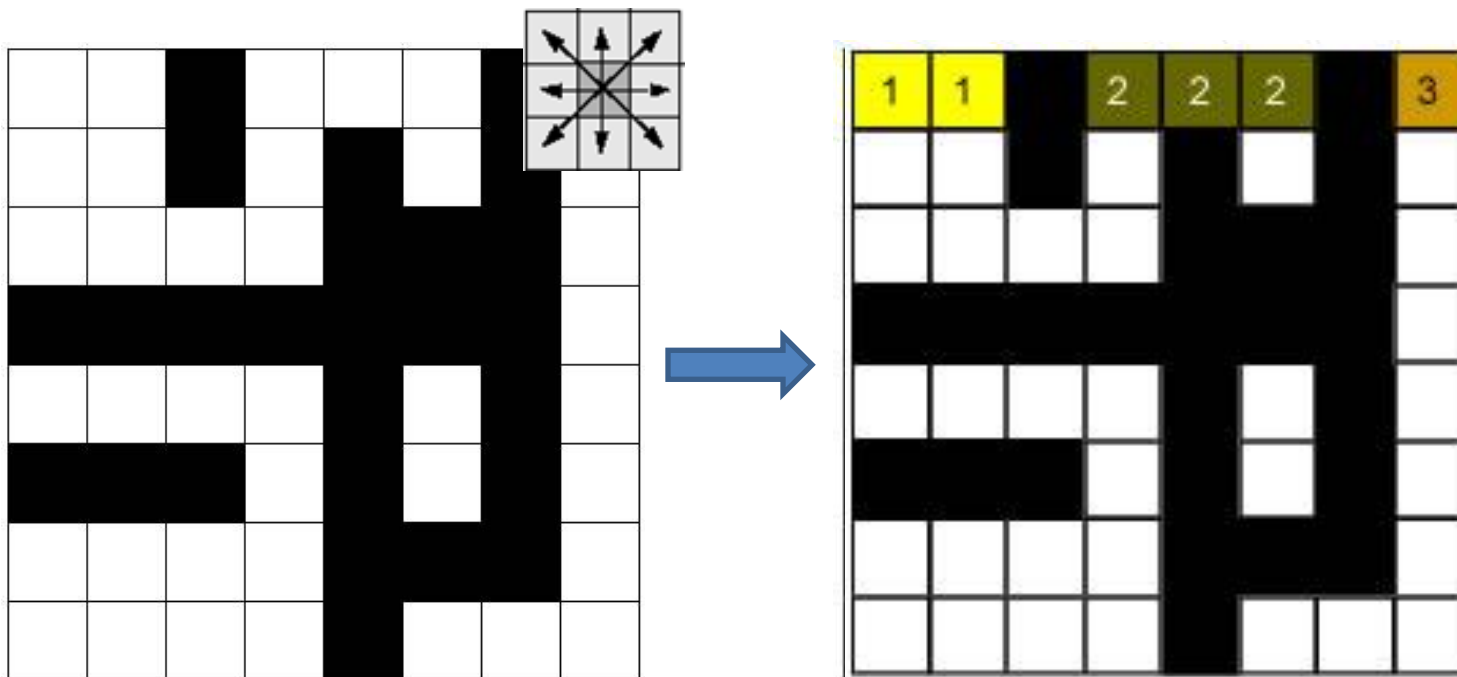
- **Pass #1:**
 - **Row #1**



Row by Row Algorithm

- Sliding a connectivity kernel , row by row (2 passes)
 - If the center falls in a non-zero pixel, label it!
 - Labeling:
 - If there are no labeled pixels connected, attribute a new label
 - Otherwise, attribute to it the neighbor's label.
 - A Union-Find structure control adjacent labels (Union-Find)

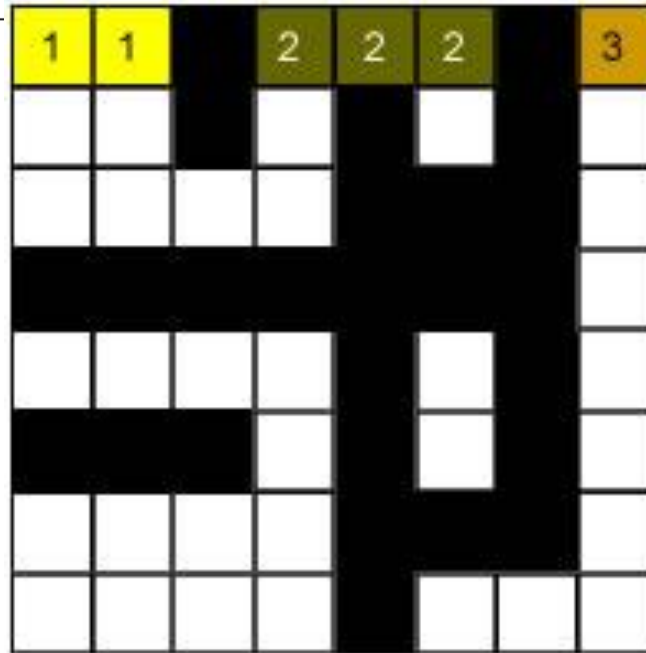
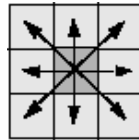
- **Pass #1:**
 - **Row #1**



Row by Row Algorithm

- Sliding a connectivity kernel , row by row (2 passes)
 - If the center falls in a non-zero pixel, label it!
 - Labeling:
 - If there are no labeled pixels connected, attribute a new label
 - Otherwise, attribute to it the neighbor 's label.
 - A Union-Find structure control adjacent labels (Union-Find)

- **Pass #1:**
 - **Row #1**

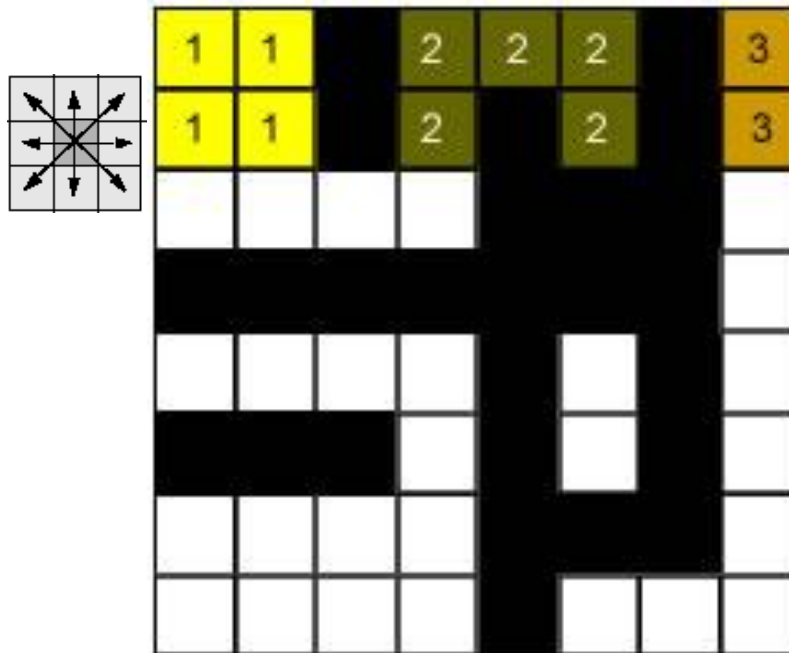


Row by Row Algorithm

- Sliding a connectivity kernel , row by row (2 passes)
 - If the center falls in a non-zero pixel, label it!
 - Labeling:
 - If there are no labeled pixels connected, attribute a new label
 - Otherwise, attribute to it the neighbor 's label.
 - A Union-Find structure control adjacent labels (Union-Find)

- **Pass #1:**

- **Row #2**

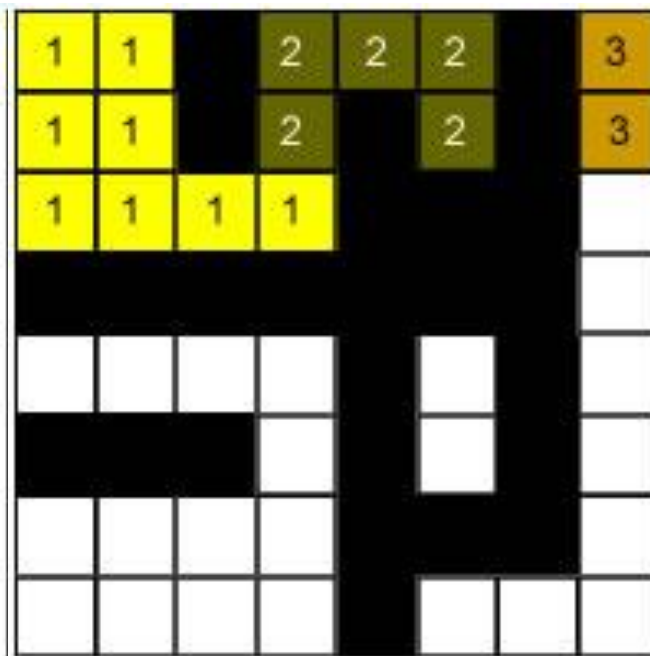
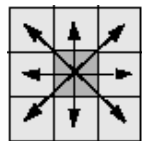


Row by Row Algorithm

- Sliding a connectivity kernel , row by row (2 passes)
 - If the center falls in a non-zero pixel, label it!
 - Labeling:
 - If there are no labeled pixels connected, attribute a new label
 - Otherwise, attribute to it the neighbor 's label.
 - A Union-Find structure control adjacent labels (Union-Find)

- Pass #1:**

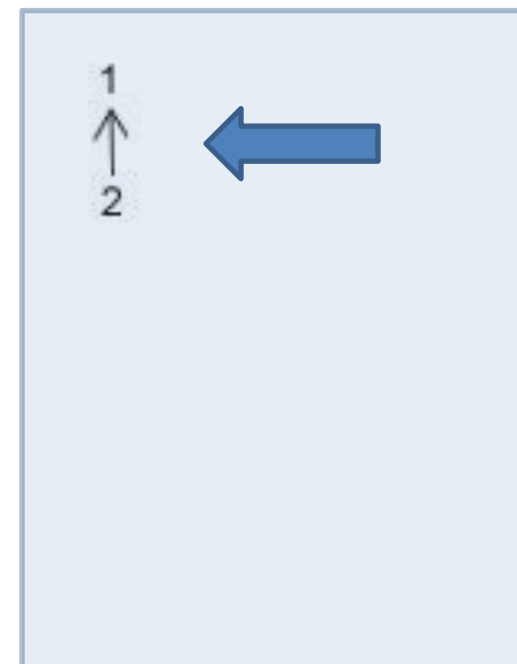
- Row #3**



Adjacent labels



Union-Find

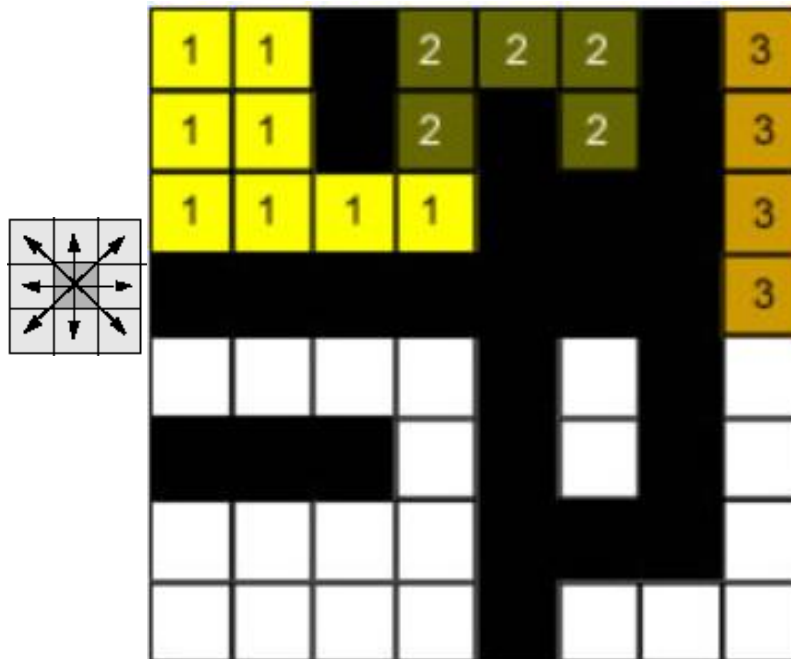


Row by Row Algorithm

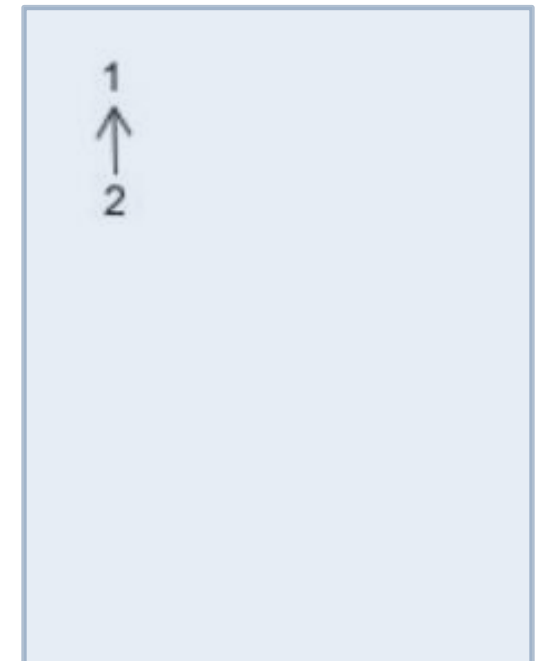
- Sliding a connectivity kernel , row by row (2 passes)
 - If the center falls in a non-zero pixel, label it!
 - Labeling:
 - If there are no labeled pixels connected, attribute a new label
 - Otherwise, attribute to it the neighbor 's label.
 - A Union-Find structure control adjacent labels (Union-Find)

- Pass #1:**

- Row #4**



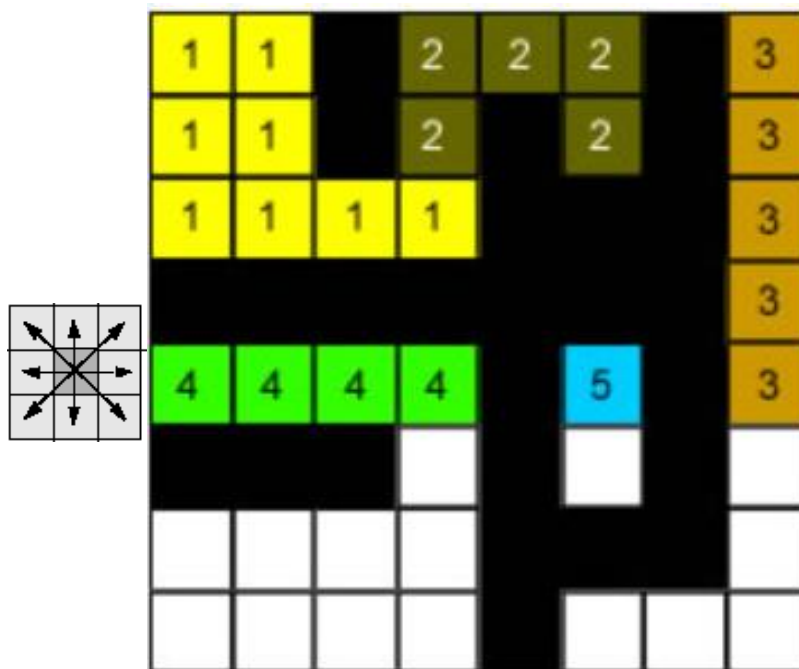
Union-Find



Row by Row Algorithm

- Sliding a connectivity kernel , row by row (2 passes)
 - If the center falls in a non-zero pixel, label it!
 - Labeling:
 - If there are no labeled pixels connected, attribute a new label
 - Otherwise, attribute to it the neighbor 's label.
 - A Union-Find structure control adjacent labels (Union-Find)

- Pass #1:**



Union-Find

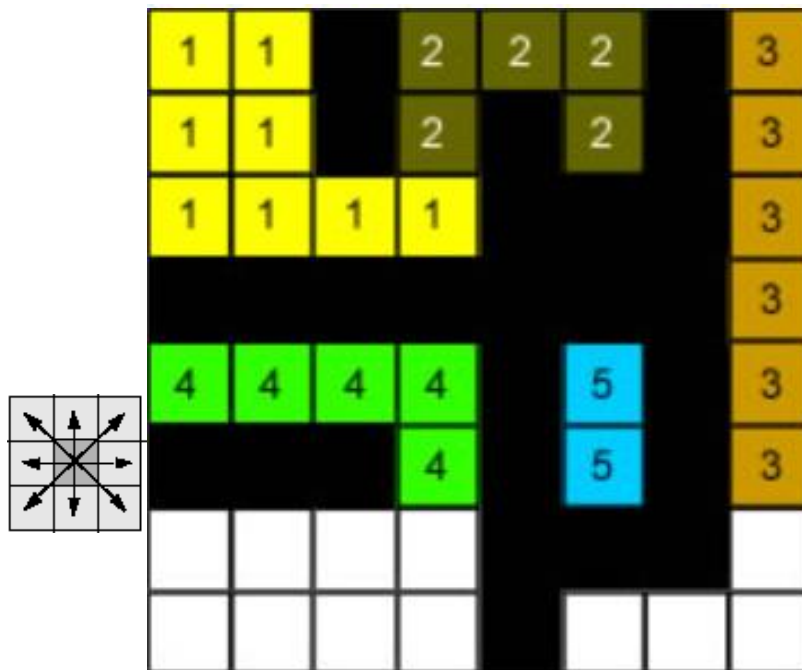


- Row #5**

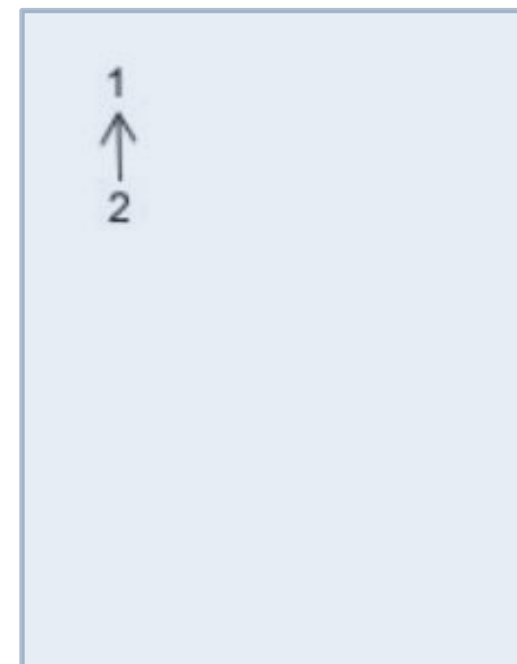
Row by Row Algorithm

- Sliding a connectivity kernel , row by row (2 passes)
 - If the center falls in a non-zero pixel, label it!
 - Labeling:
 - If there are no labeled pixels connected, attribute a new label
 - Otherwise, attribute to it the neighbor 's label.
 - A Union-Find structure control adjacent labels (Union-Find)

- **Pass #1:**



Union-Find

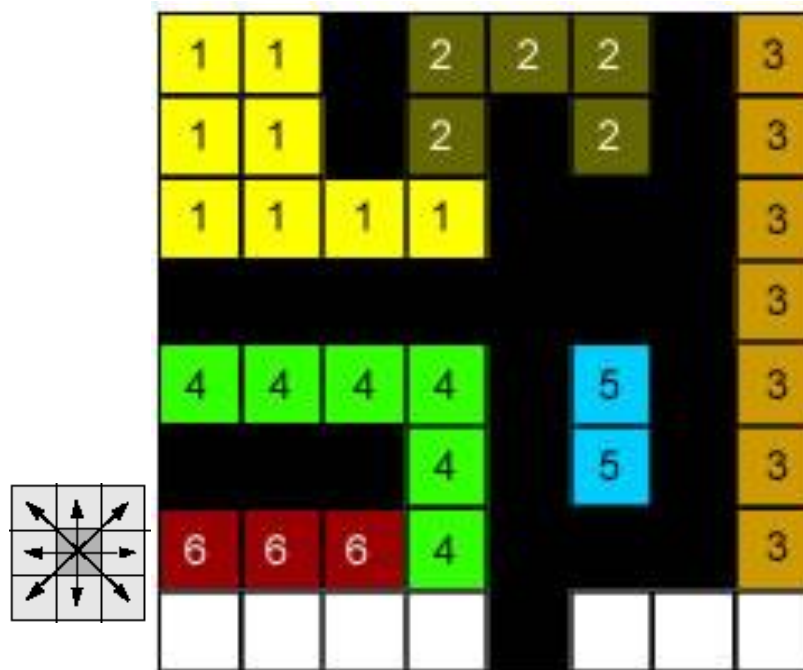


- **Row #6**

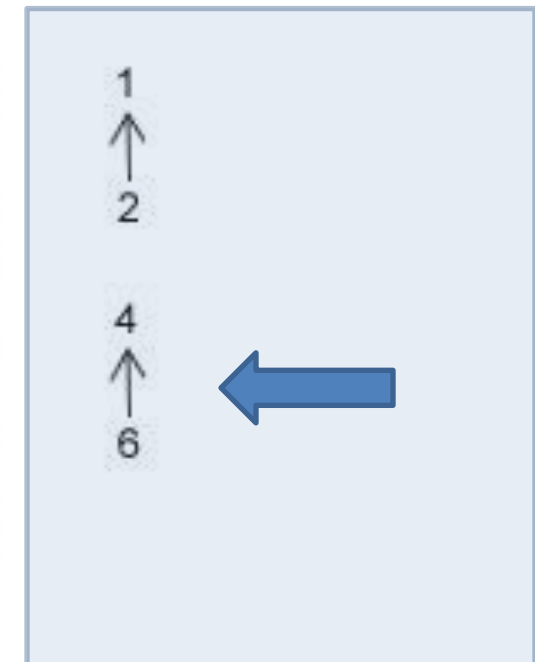
Row by Row Algorithm

- Sliding a connectivity kernel , row by row (2 passes)
 - If the center falls in a non-zero pixel, label it!
 - Labeling:
 - If there are no labeled pixels connected, attribute a new label
 - Otherwise, attribute to it the neighbor 's label.
 - A Union-Find structure control adjacent labels (Union-Find)

- Pass #1:**



Union-Find

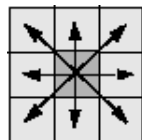
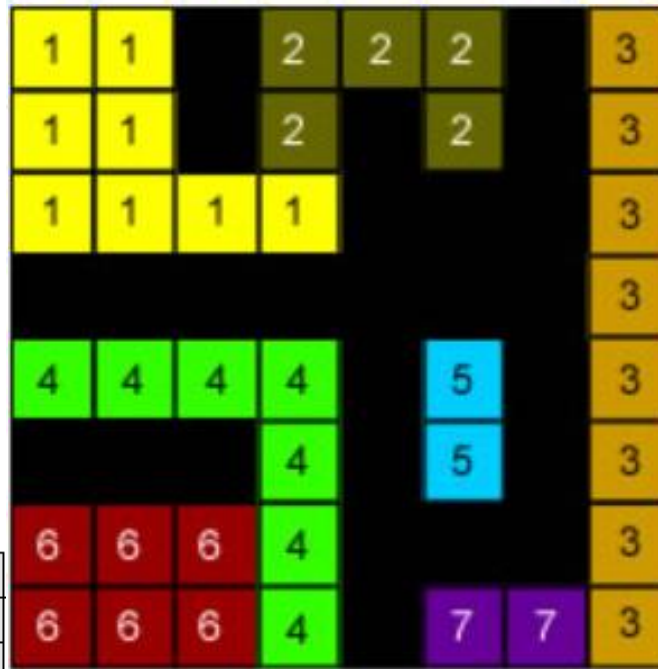


- Row #7**

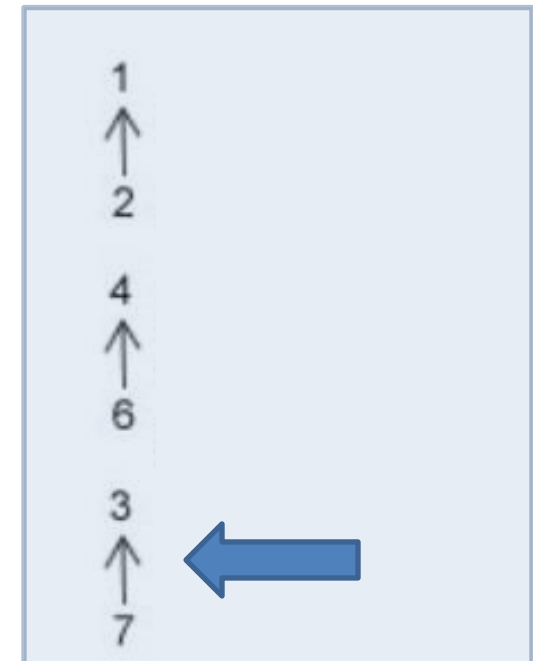
Row by Row Algorithm

- Sliding a connectivity kernel , row by row (2 passes)
 - If the center falls in a non-zero pixel, label it!
 - Labeling:
 - If there are no labeled pixels connected, attribute a new label
 - Otherwise, attribute to it the neighbor 's label.
 - A Union-Find structure control adjacent labels (Union-Find)

- **Pass #1:**



Union-Find



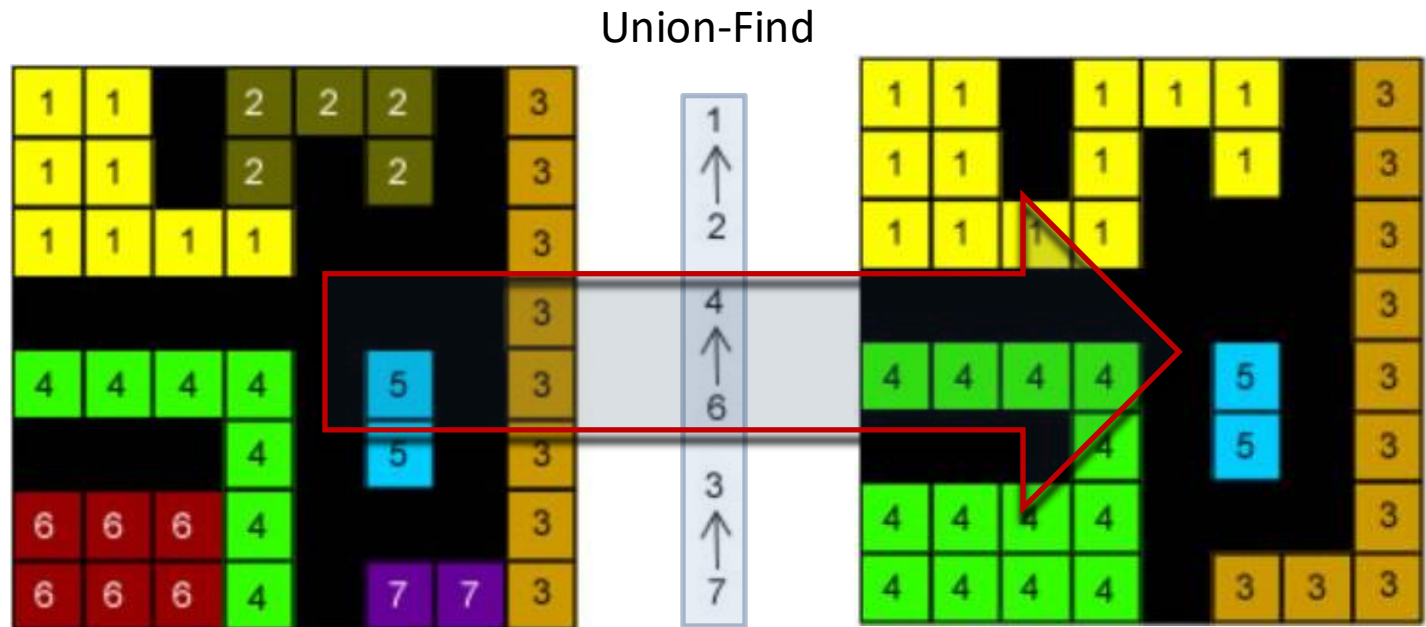
- **Row #8**

Row by Row Algorithm

- Sliding a connectivity kernel , row by row (2 passes)
 - If the center falls in a non-zero pixel, label it!
 - Labeling:
 - If there are no labeled pixels connected, attribute a new label
 - Otherwise, attribute to it the neighbor 's label.
 - A Union-Find structure control adjacent labels (Union-Find)

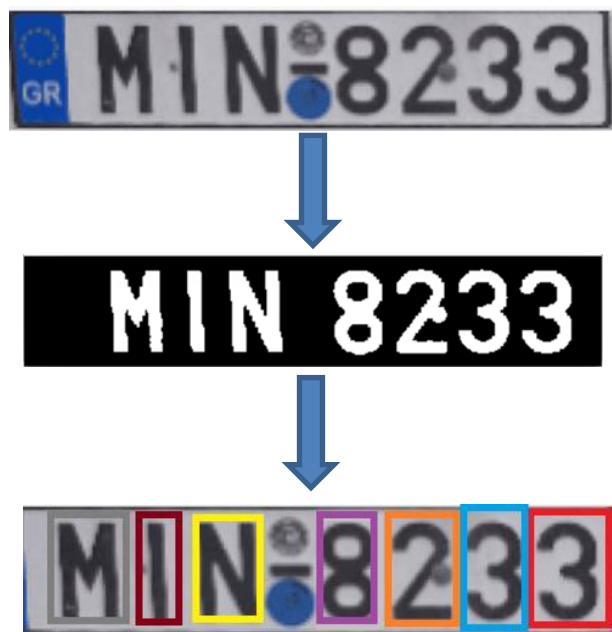
- Pass #2:**

- Resolve Union-Find**



Let's Code!

- In our practice, we will implement an algorithm to segment characters in a license plate.



- Besides, we will introduce the `cv2.connectedComponent()` that implements the component labeling method
- Checkout it here: [Lecture 04 - Finding Components.ipynb](#)