Slide 2

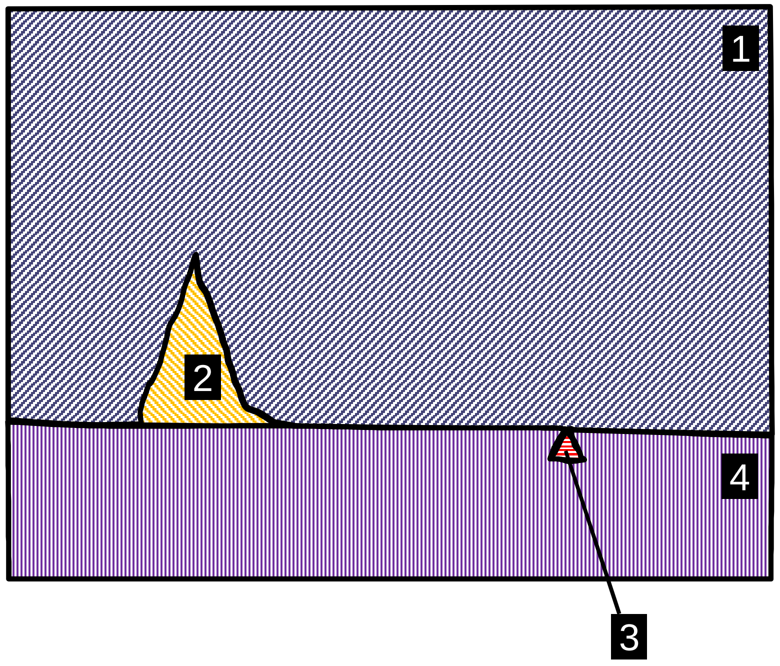
Segmentation

- partitioning the image into areas of similar color
  - image driven
  - no semantics for segments

- what we need for segmentation:
  - a criterion that defines which pixels belong to a segment and which don’t
  - an algorithm that efficiently subdivides pixels into segments

Segmentation

- partitioning the image into areas of similar color
  - image driven
  - no semantics for segments
- what we need for segmentation:
  - a criterion that defines which pixels belong to a segment and which don't
  - an algorithm that efficiently subdivides pixels into segments

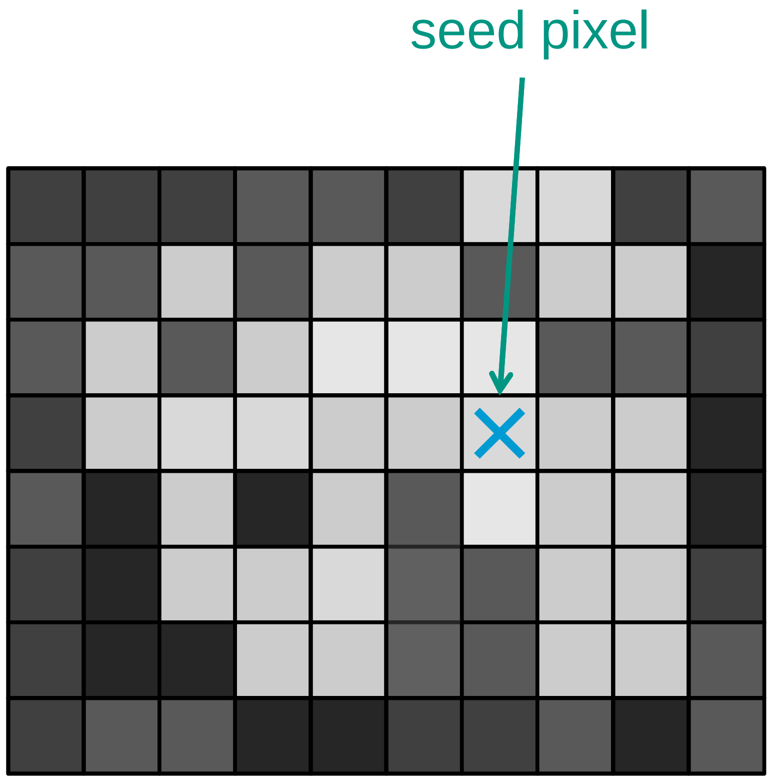


Slide 12

Region Growing

– key idea:
  - start from one/more seed points (seed points must be provided)
  - incrementally expand segment until any pixel can be added
  - implements connectedness criterion + homogeneity or neighborhood criterion
  - yields single segment

– advantages and disadvantages:
  - easy to implement (breadth-first-search)
  - requires one or more seed points

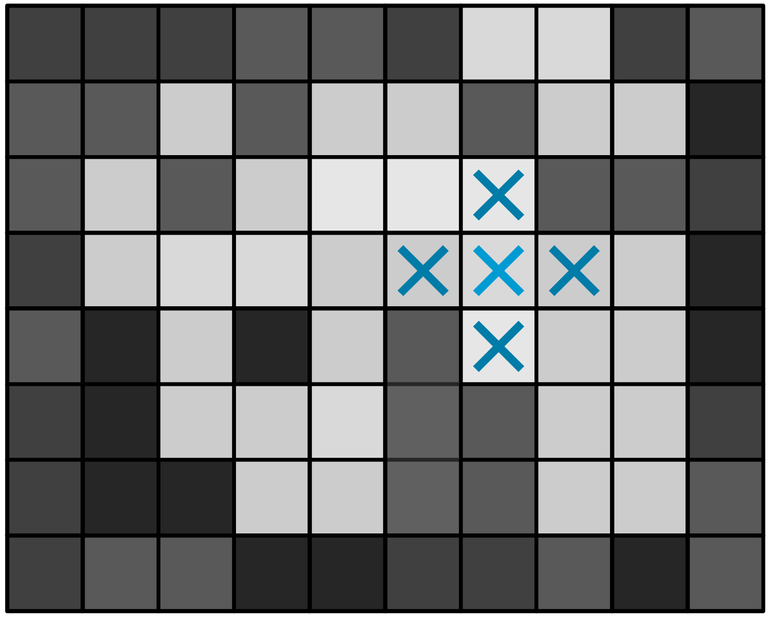


Region Growing

– key idea:
  - start from one/more seed points (seed points must be provided)
  - incrementally expand segment until any pixel can be added

  - implements connectedness criterion + homogeneity or neighborhood criterion
  - yields single segment

– advantages and disadvantages:
  - easy to implement (breadth-first-search)
  - requires one or more seed points

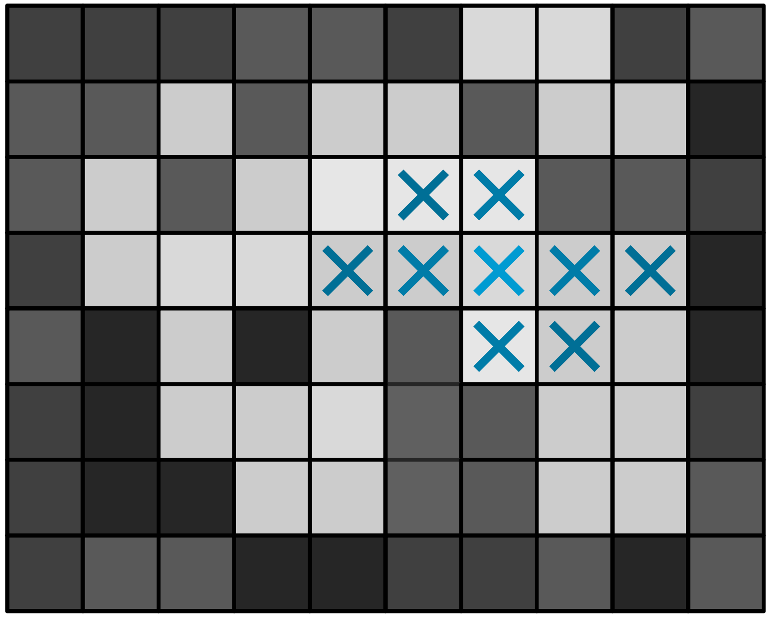


Region Growing

– key idea:
  - start from one/more seed points (seed points must be provided)
  - incrementally expand segment until any pixel can be added

  - implements connectedness criterion + homogeneity or neighborhood criterion
  - yields single segment

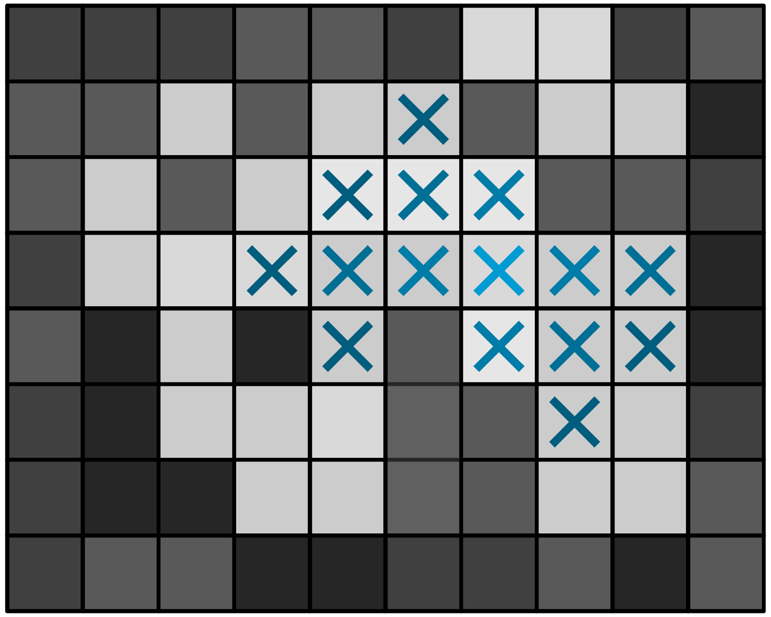
– advantages and disadvantages:
  - easy to implement (breadth-first-search)
  - requires one or more seed points



Region Growing

– key idea:
  - start from one/more seed points (seed points must be provided)
  - incrementally expand segment until any pixel can be added
  - implements connectedness criterion + homogeneity or neighborhood criterion
  - yields single segment

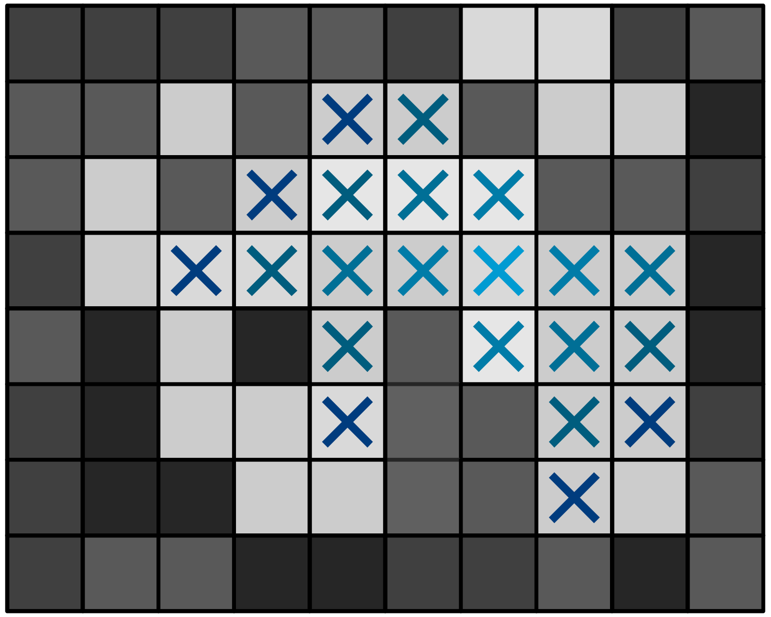
– advantages and disadvantages:
  - easy to implement (breadth-first-search)
  - requires one or more seed points



Region Growing

– key idea:
  - start from one/more seed points (seed points must be provided)
  - incrementally expand segment until any pixel can be added
  - implements connectedness criterion + homogeneity or neighborhood criterion
  - yields single segment

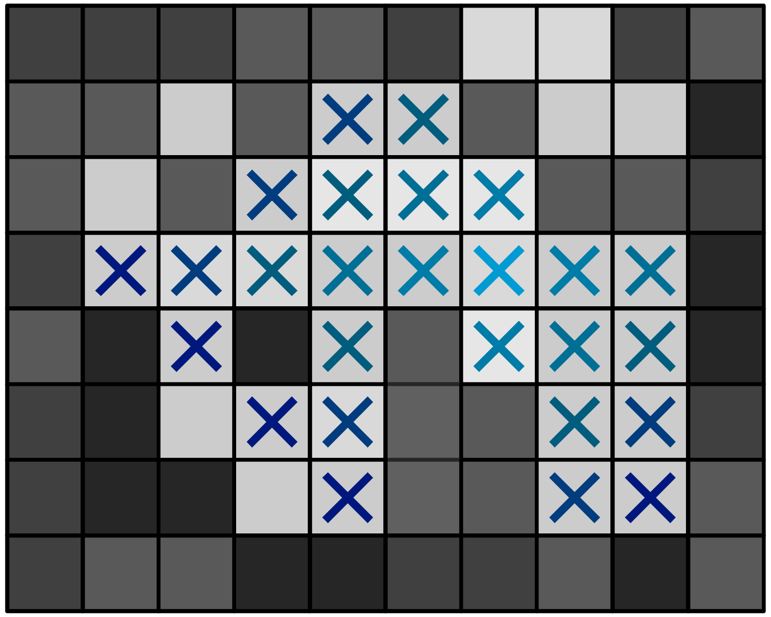
– advantages and disadvantages:
  - easy to implement (breadth-first-search)
  - requires one or more seed points



Region Growing

– key idea:
  - start from one/more seed points (seed points must be provided)
  - incrementally expand segment until any pixel can be added
  - implements connectedness criterion + homogeneity or neighborhood criterion
  - yields single segment

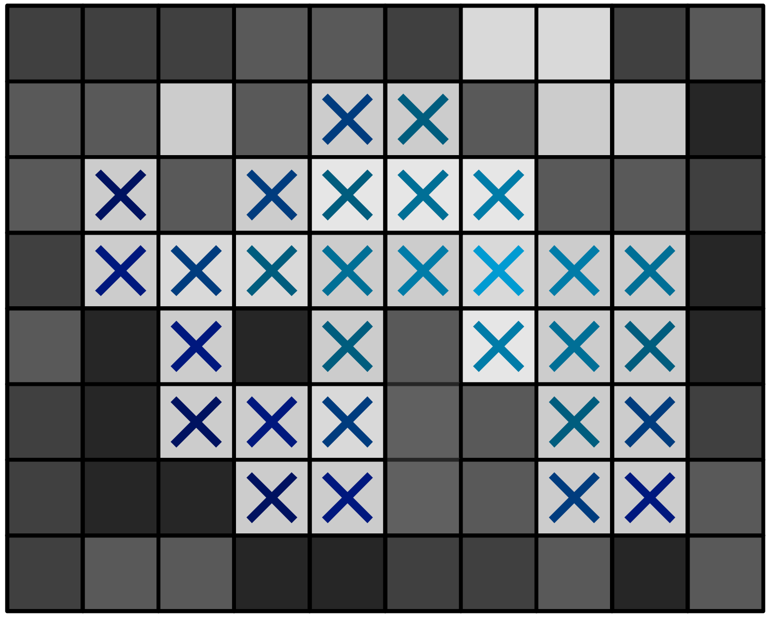
– advantages and disadvantages:
  - easy to implement (breadth-first-search)
  - requires one or more seed points



Region Growing

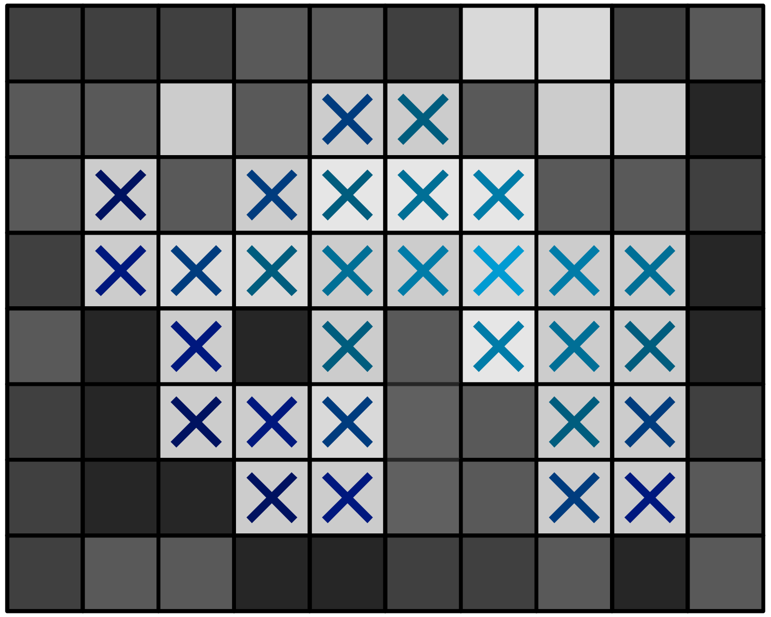
– key idea:
  - start from one/more seed points (seed points must be provided)
  - incrementally expand segment until any pixel can be added
  - implements connectedness criterion + homogeneity or neighborhood criterion
  - yields single segment

– advantages and disadvantages:
  - easy to implement (breadth-first-search)
  - requires one or more seed points



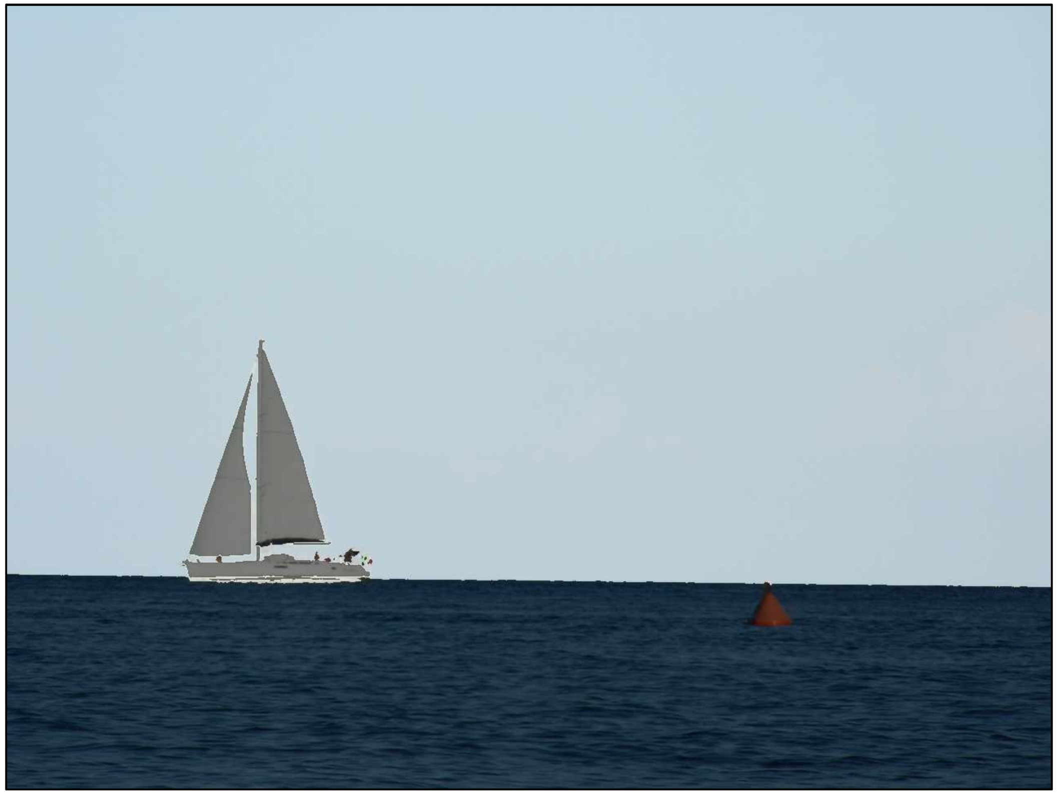
Region Growing

- key idea:
  - start from one/more seed points (seed points must be provided)
  - incrementally expand segment until any pixel can be added
  - implements connectedness criterion + homogeneity or neighborhood criterion
  - yields single segment
- advantages and disadvantages:
  - easy to implement (breadth-first-search)
  - requires one or more seed points


no more extension possible

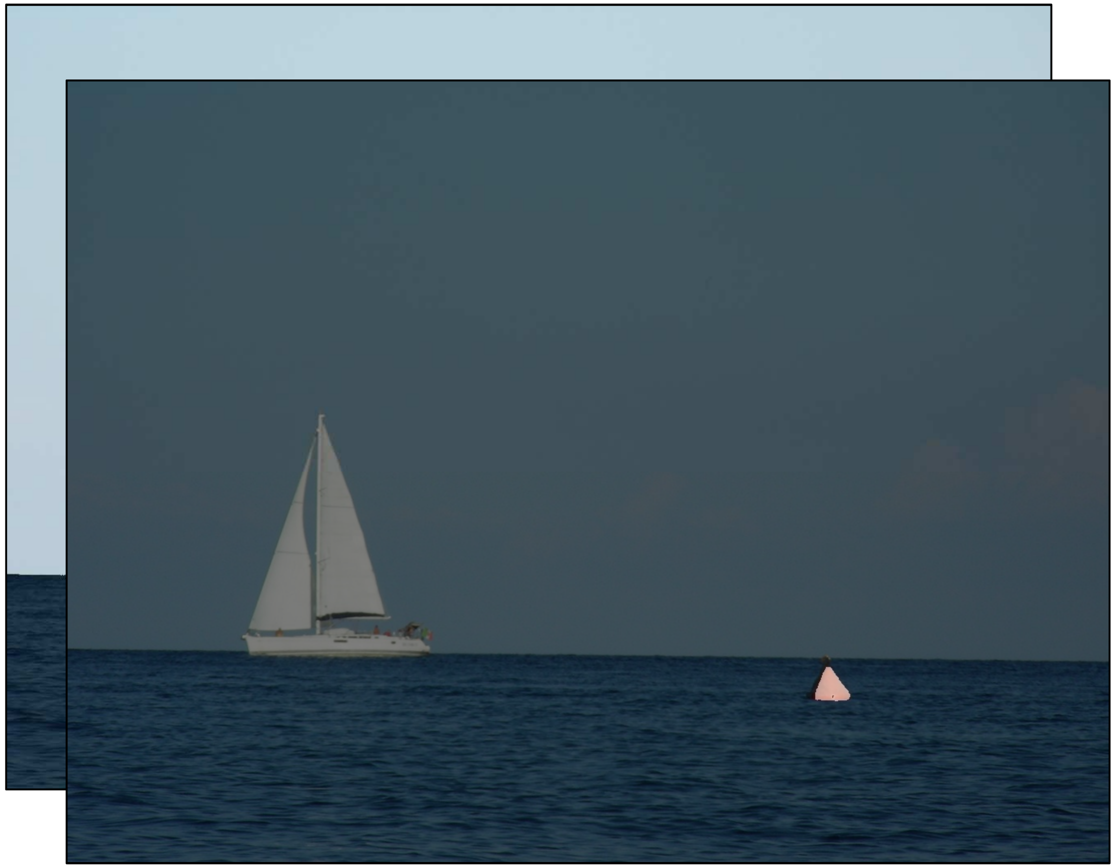
Slide 13

Region Growing cont.



region growing on RGB

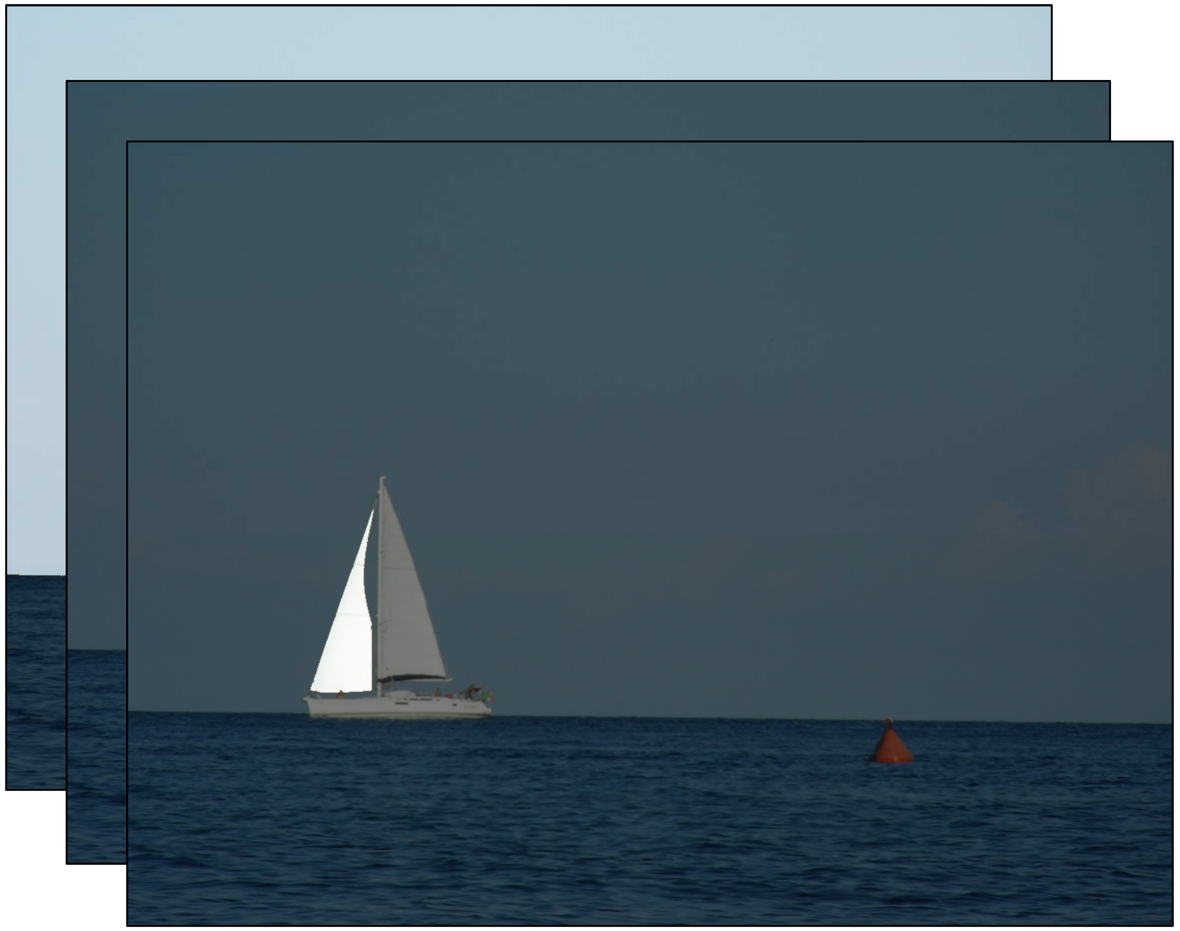
Region Growing cont.



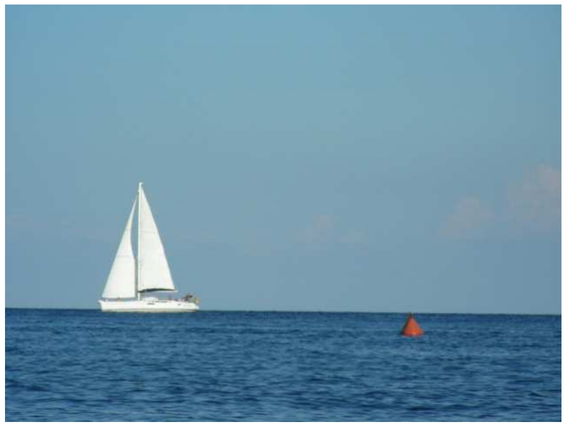
region growing on RGB

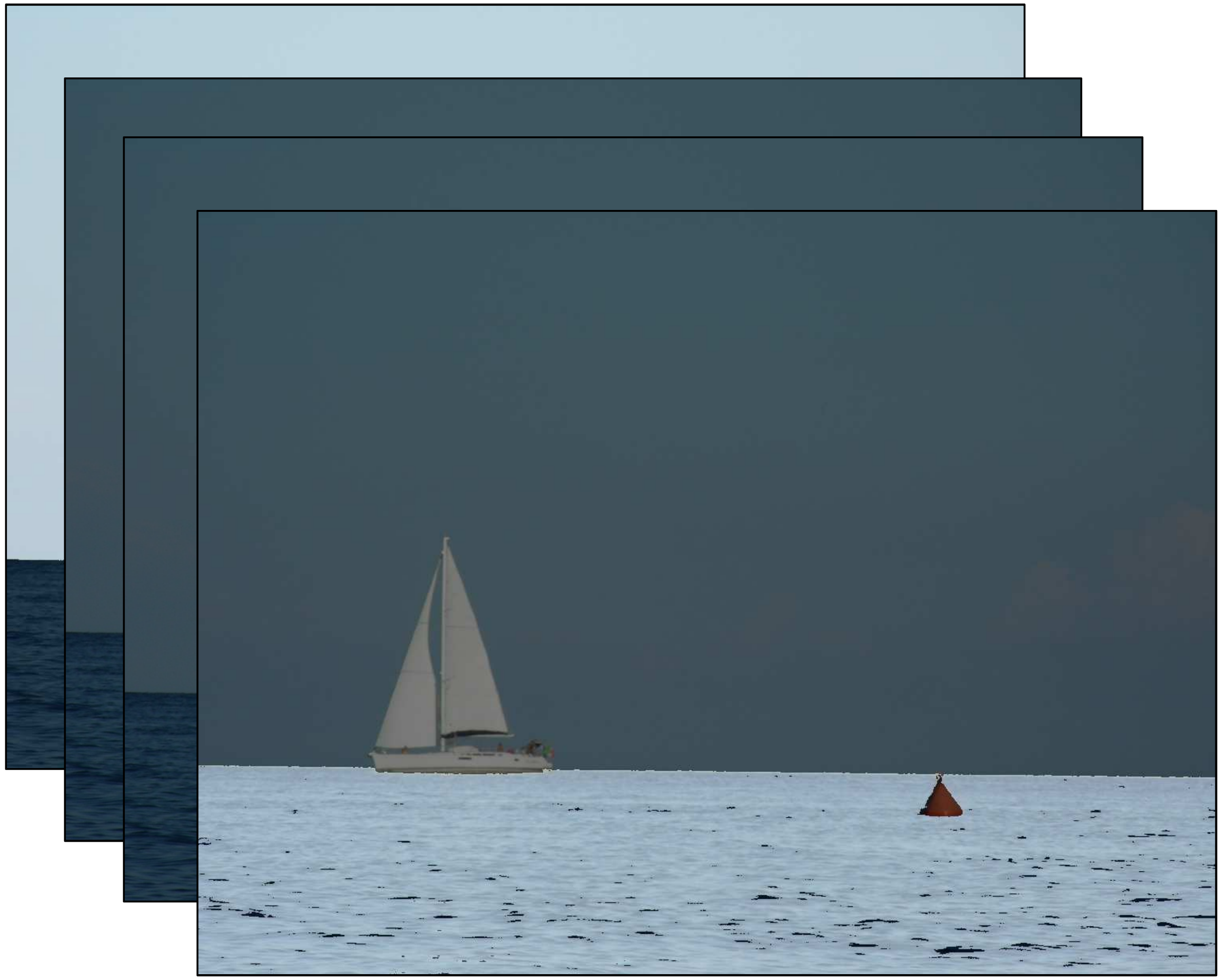
Region Growing cont.

region growing on RGB



Region Growing cont.



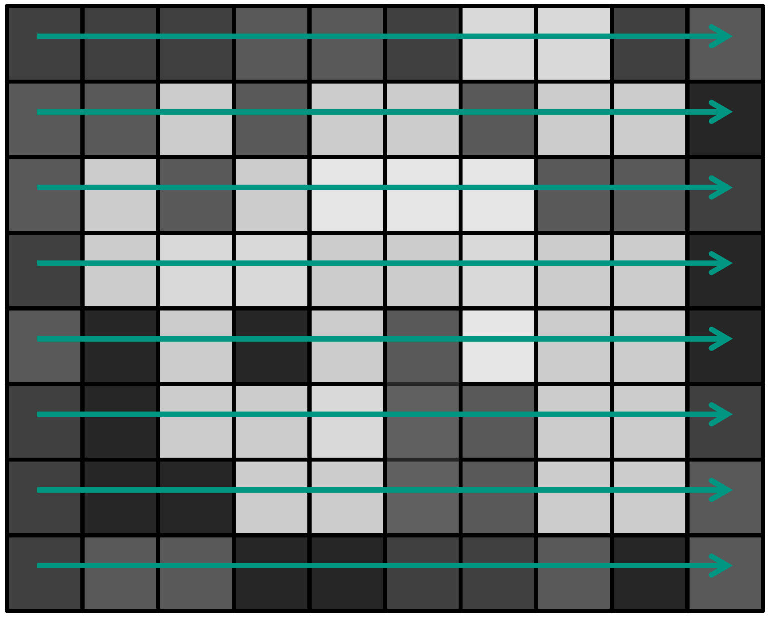


region growing on RGB

Slide 15

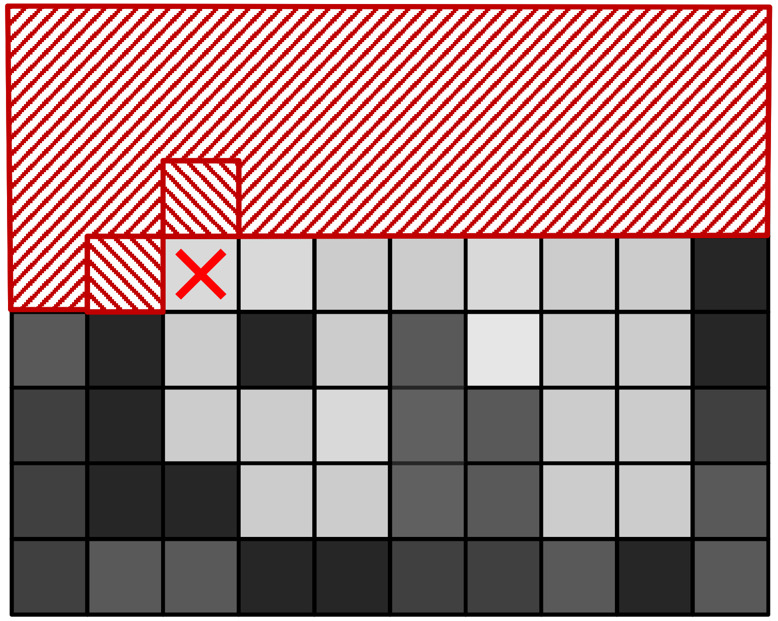
Connected Components Labeling (CCL)

– procedure:
  - we visit pixels row-by-row from the left upper corner to the right lower corner and immediately assign them to a segment



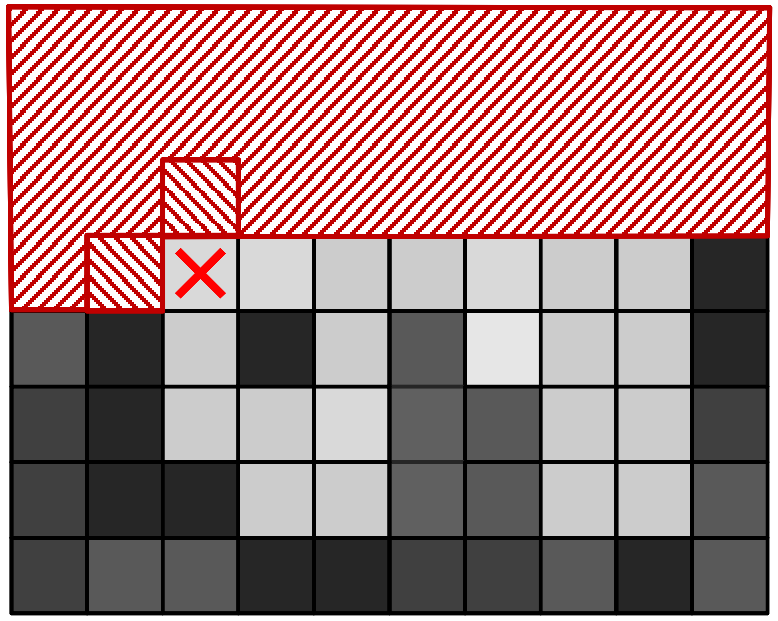
Connected Components Labeling (CCL)

– procedure:
  - we visit pixels row-by-row from the left upper corner to the right lower corner and immediately assign them to a segment
  - when we visit a pixel (u,v) we already visited (u-1,v) and (u,v-1)



Connected Components Labeling (CCL)

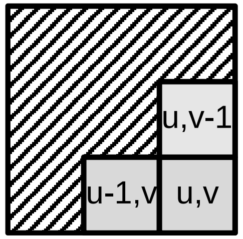
– procedure:
  - we visit pixels row-by-row from the left upper corner to the right lower corner and immediately assign them to a segment
  - when we visit a pixel (u,v) we already visited (u-1,v) and (u,v-1)
  - we compare color(u,v) with color(u-1,v), color(u,v-1). Five cases



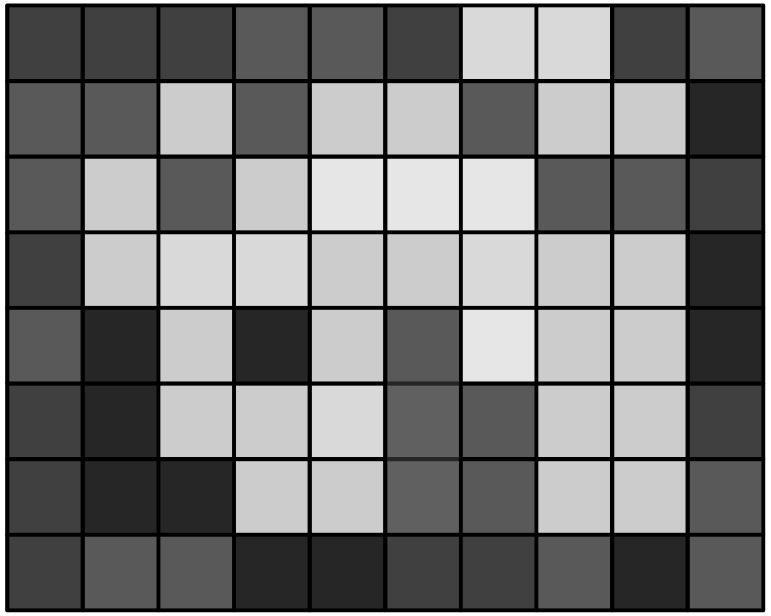
Slide 17

Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

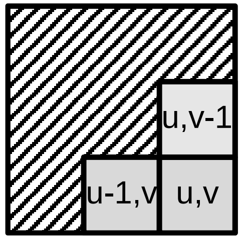


- Example

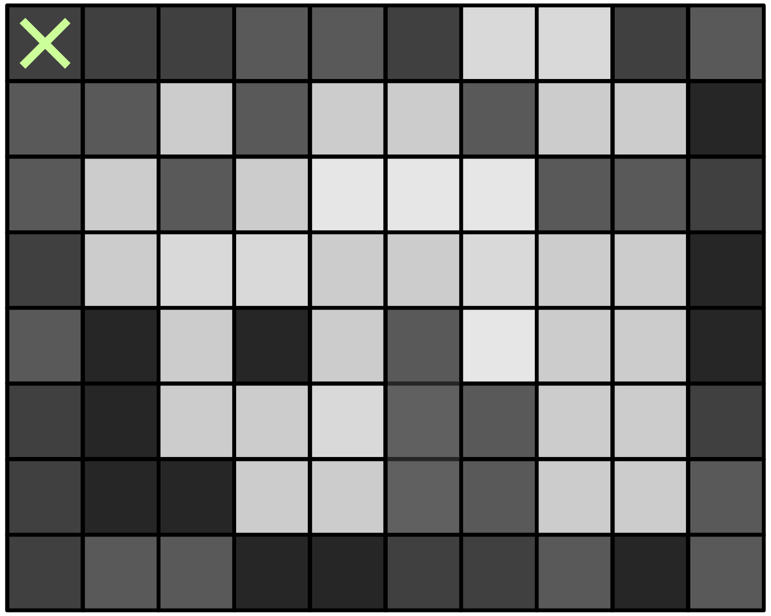


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

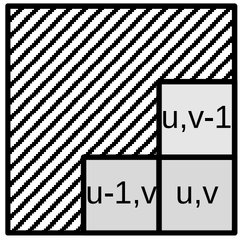


- Example

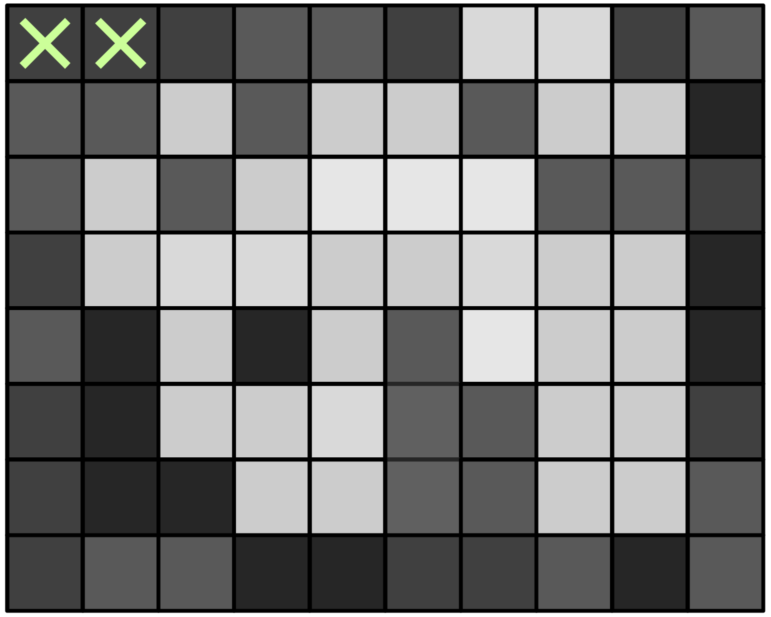


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

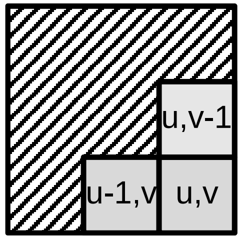


- Example

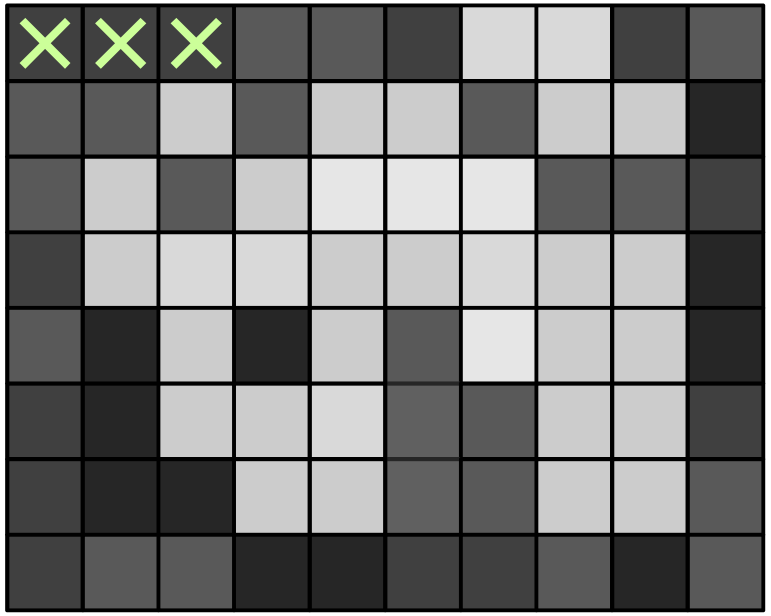


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

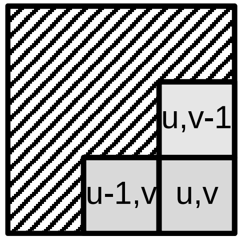


- Example

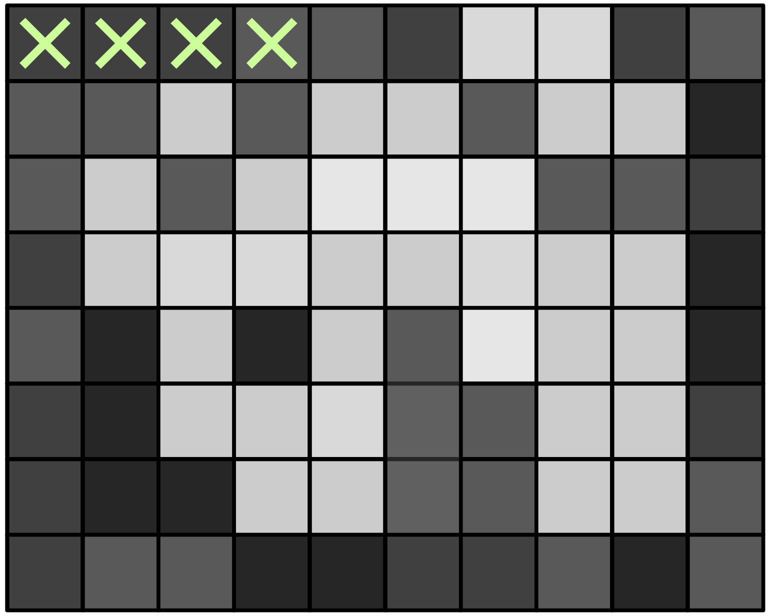


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

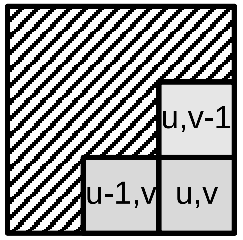


- Example

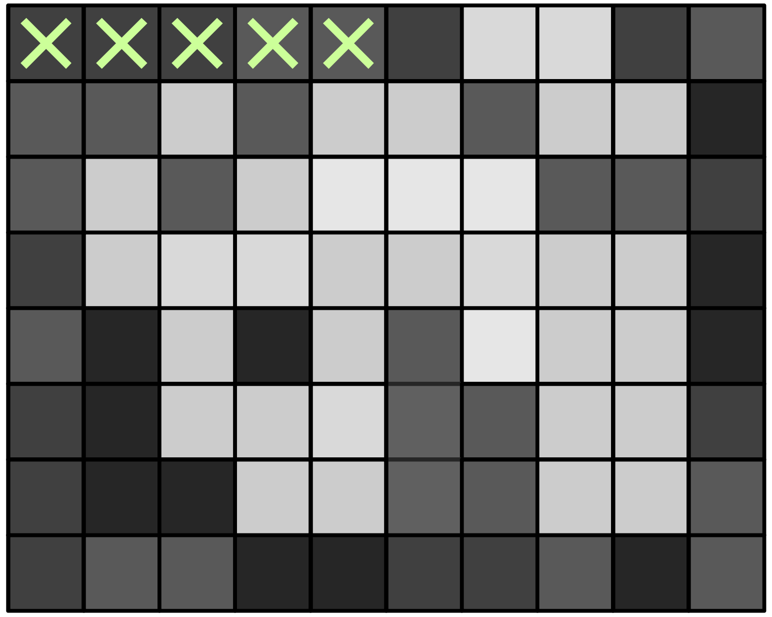


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

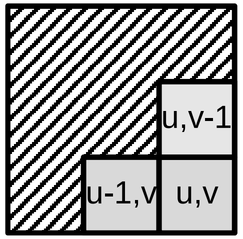


- Example

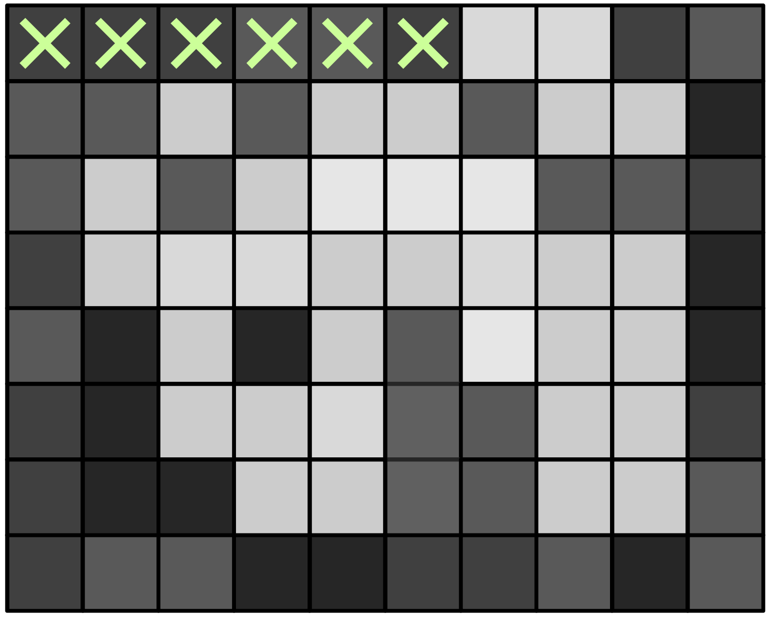


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

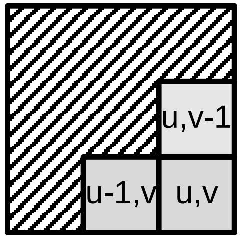


- Example

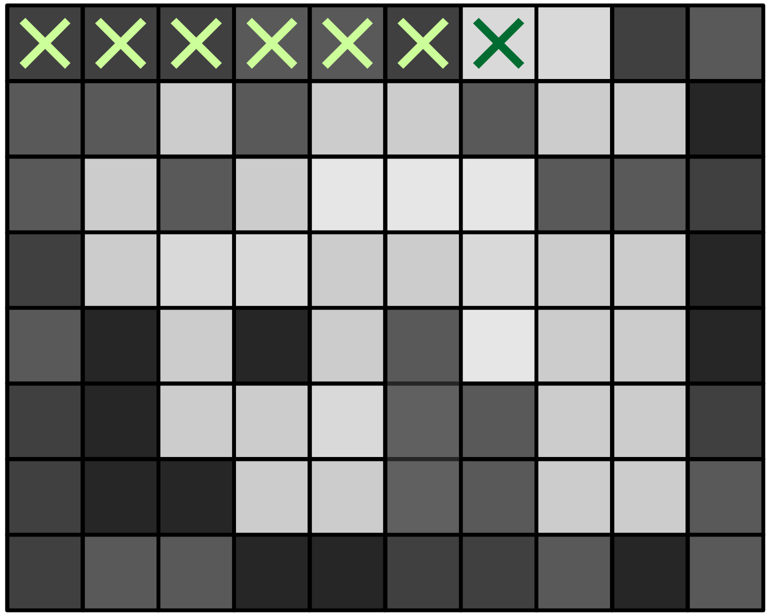


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

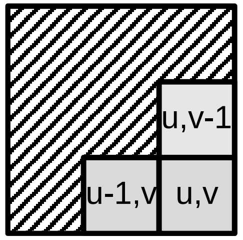


- Example

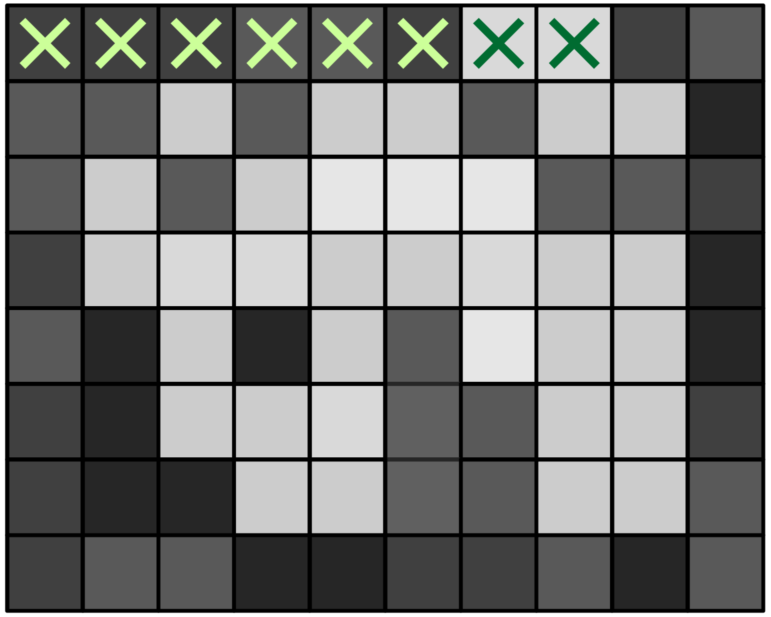


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment



- Example



Connected Components Labeling (CCL)

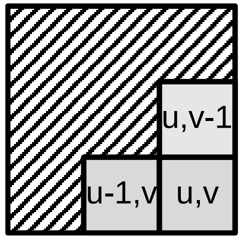
5. pixel colors at (u,v) and (u-1,v) are similar

   pixel colors at (u,v) and (u,v-1) are similar

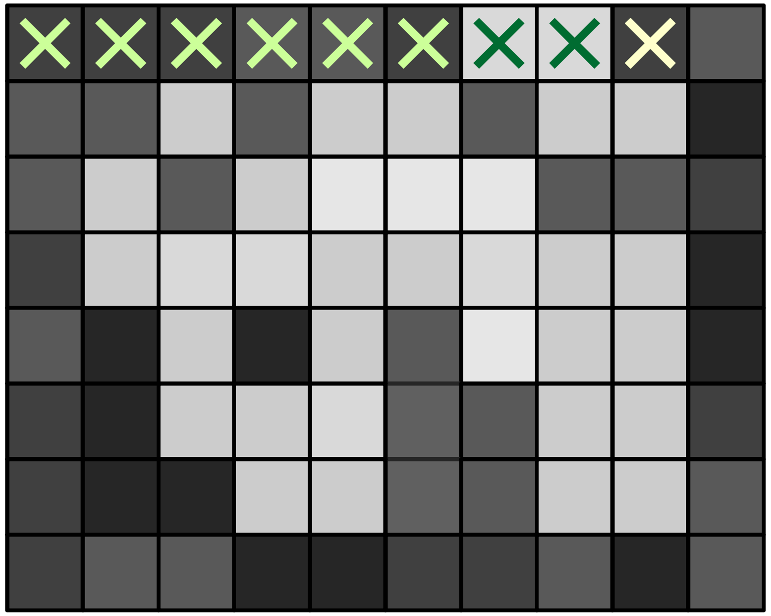
   pixels (u-1,v) and (u,v-1) do not belong to the same segment

   → pixel (u,v) belongs to the segments of both neighbors

   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

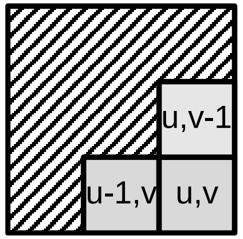


- Example

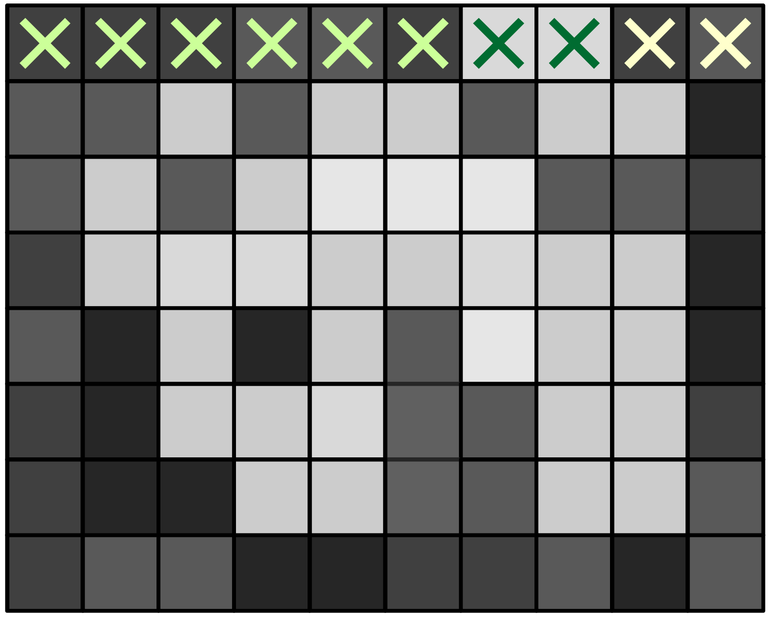


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

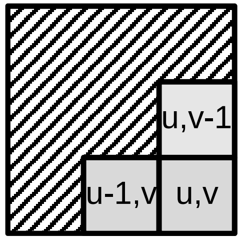


- Example

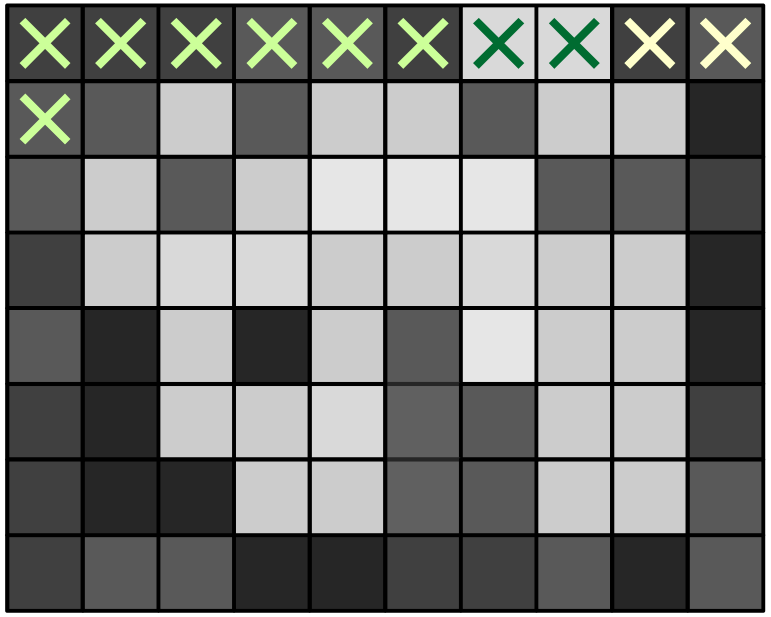


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

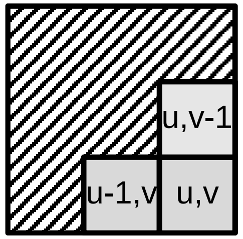


- Example

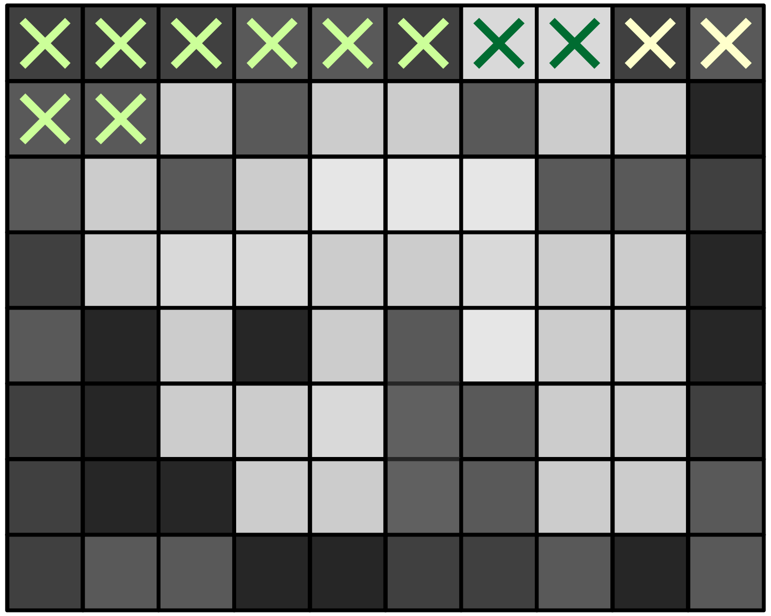


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

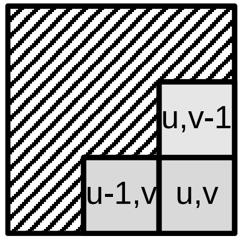


- Example

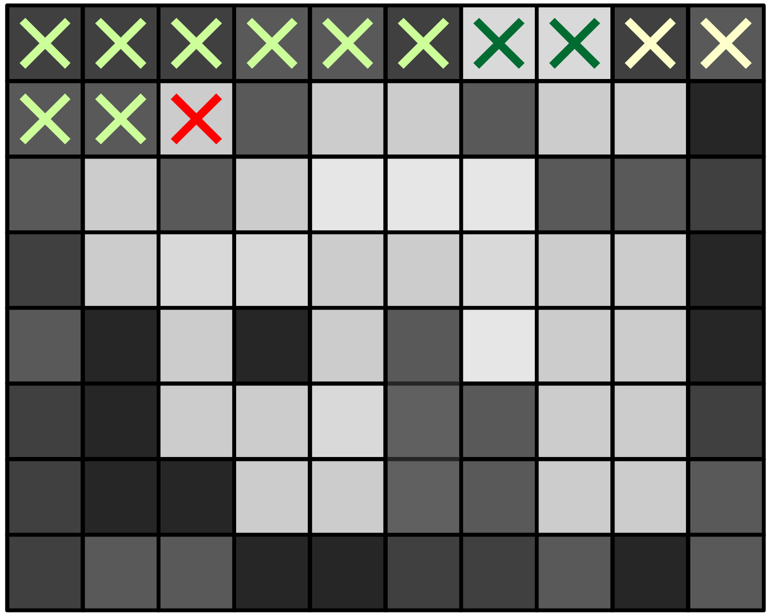


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

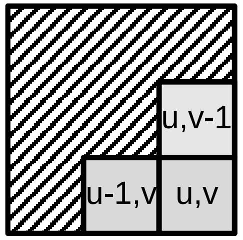


- Example

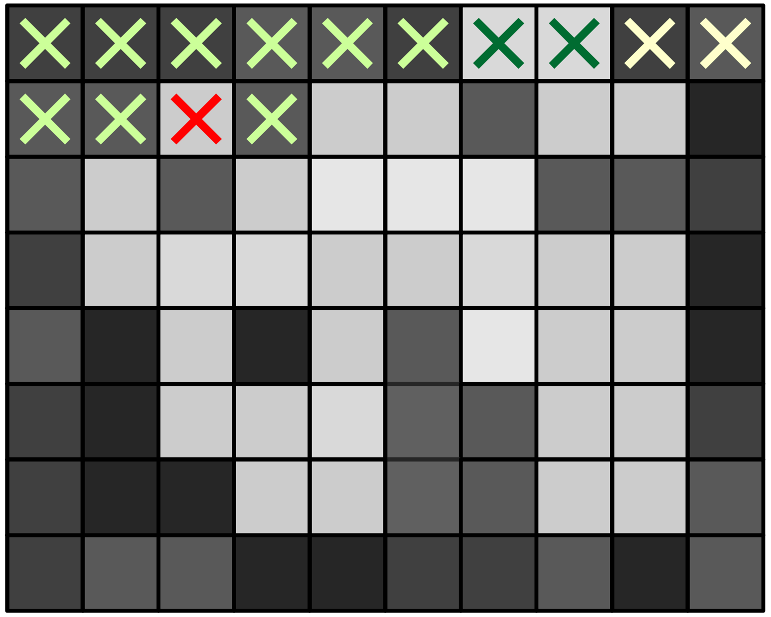


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

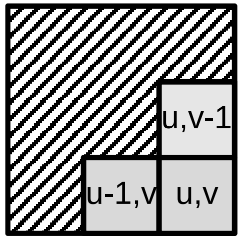


- Example

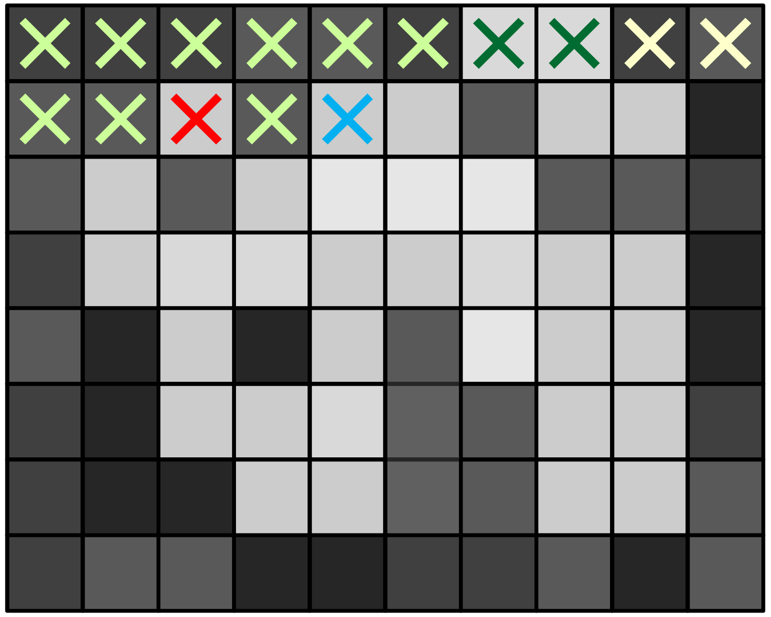


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

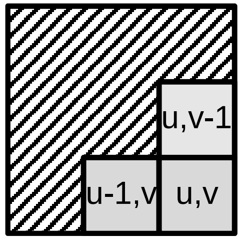


- Example

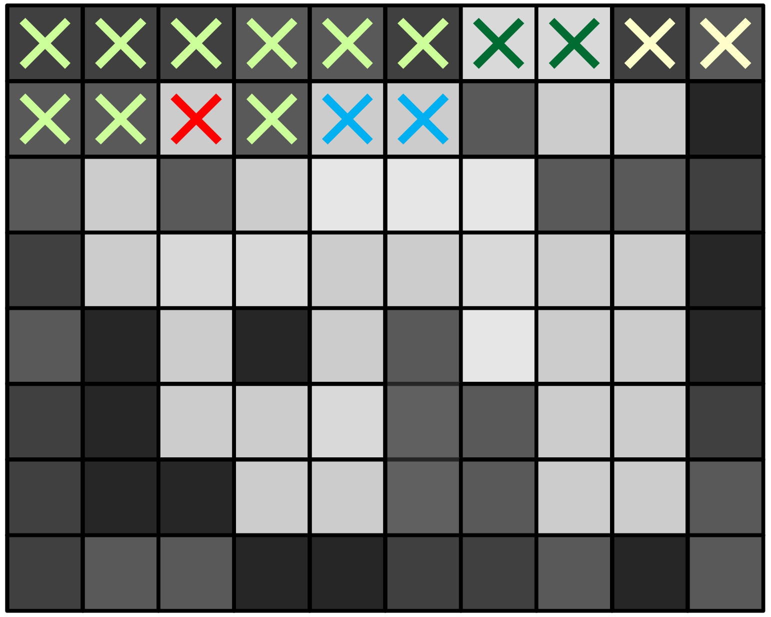


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

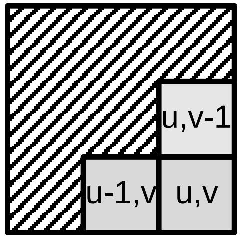


- Example

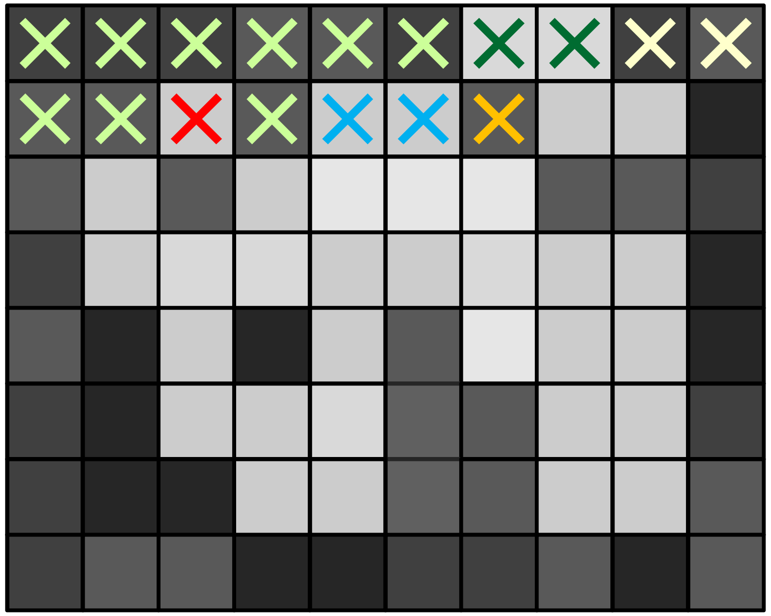


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

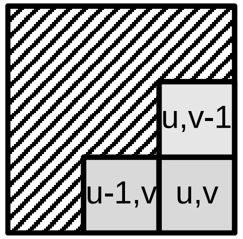


- Example

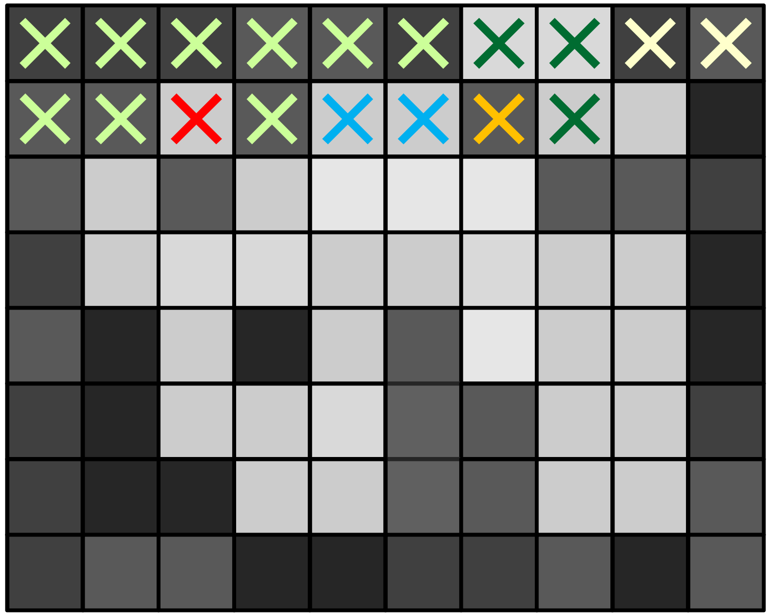


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

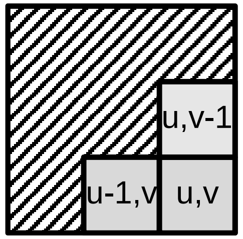


- Example

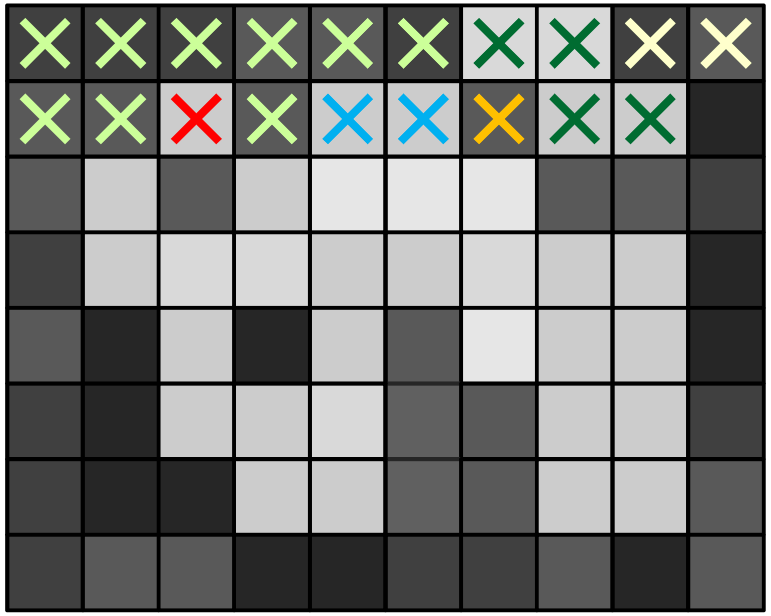


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

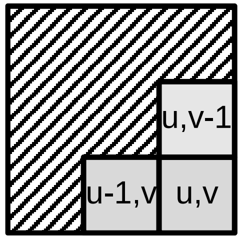


- Example

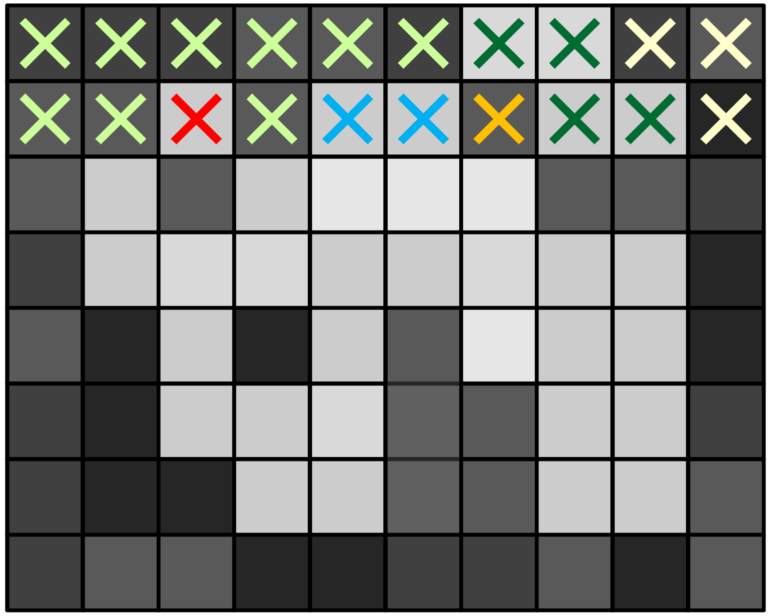


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

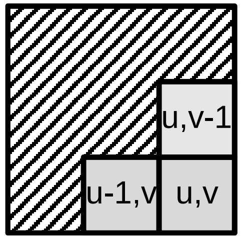


- Example

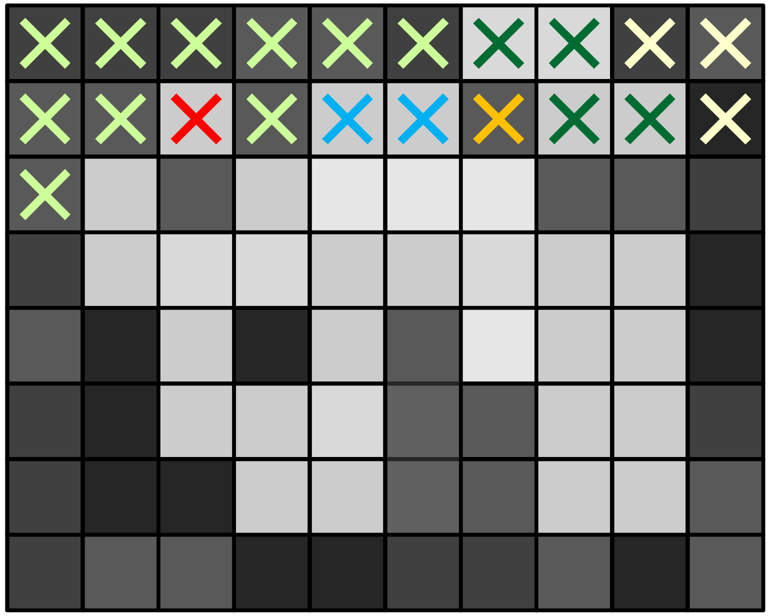


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

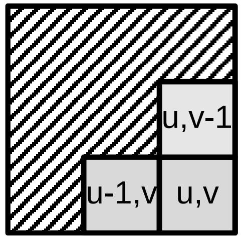


- Example

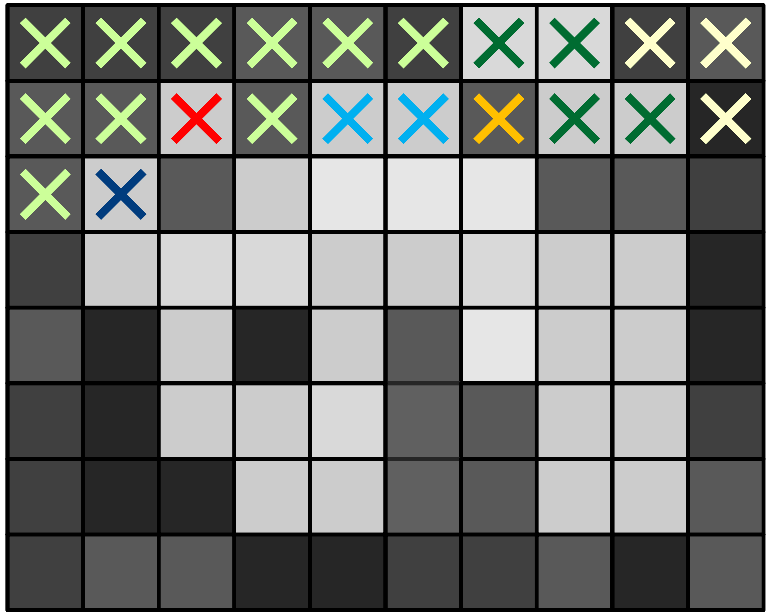


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

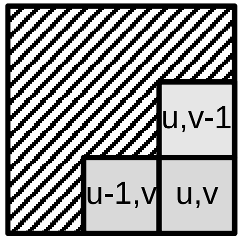


- Example

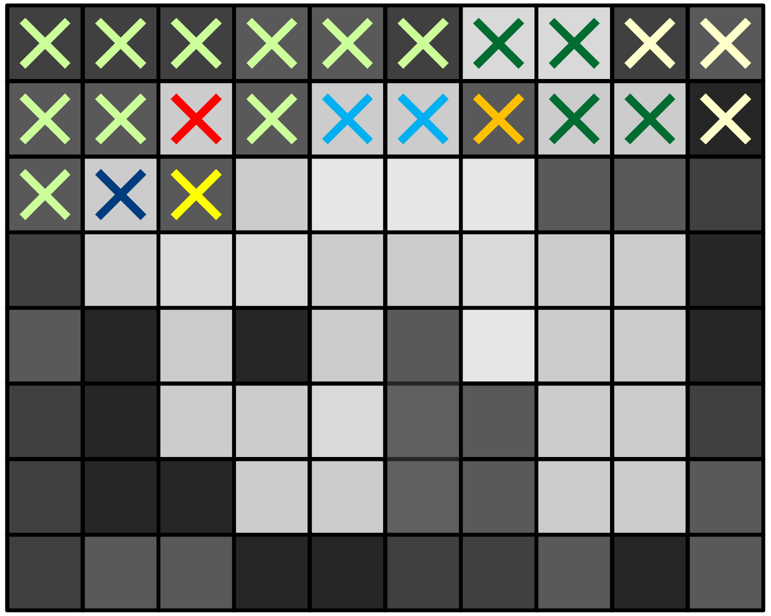


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

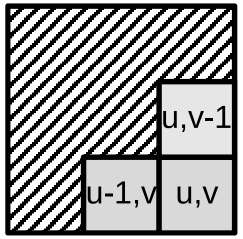


- Example

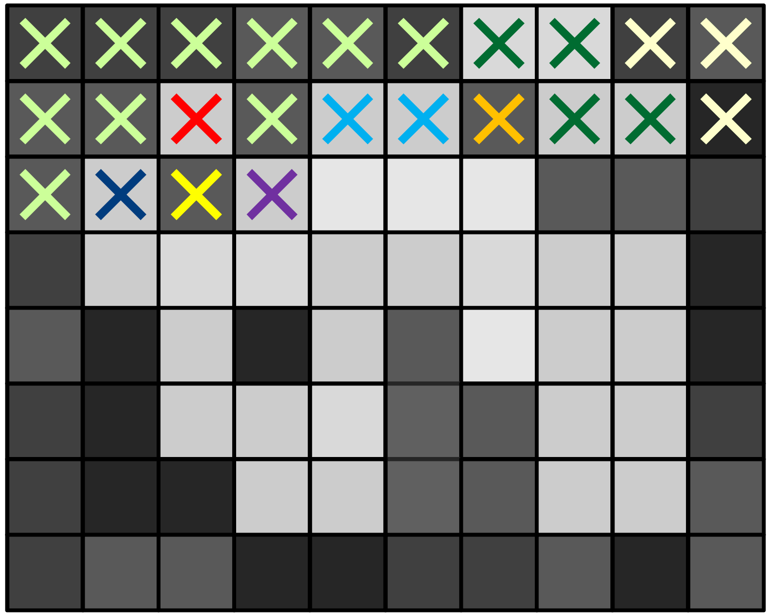


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

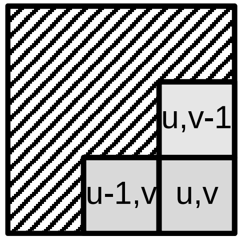


- Example

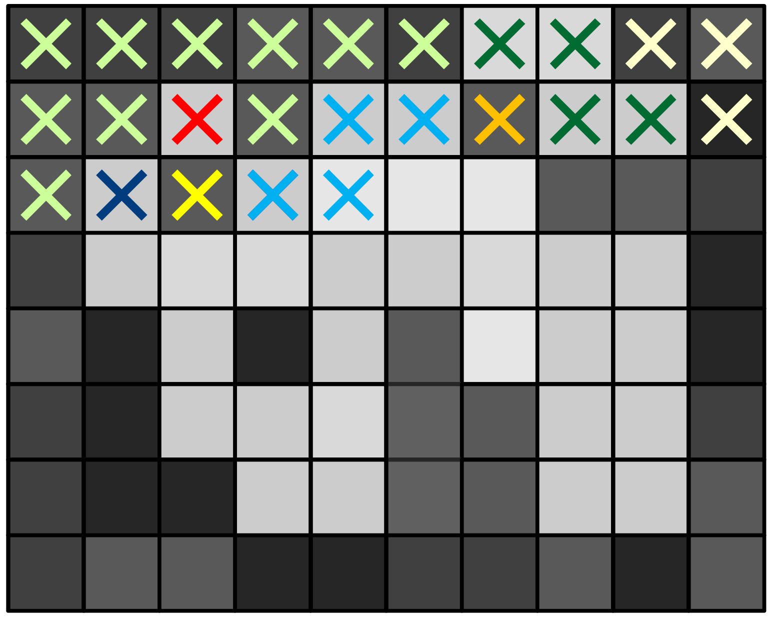


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

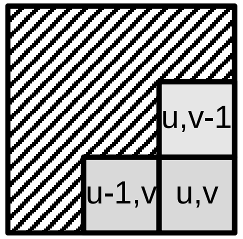


- Example

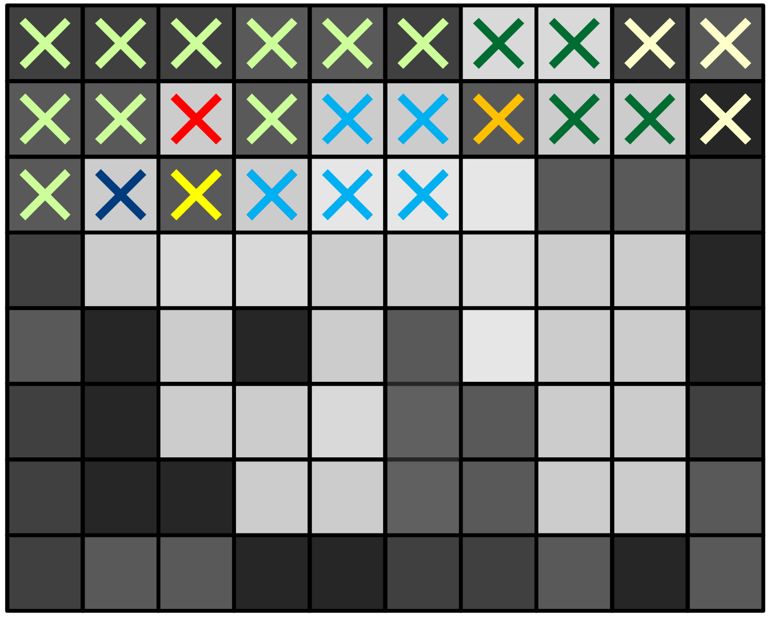


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

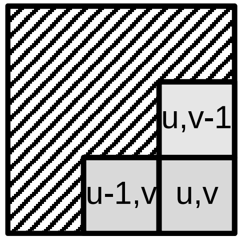


- Example

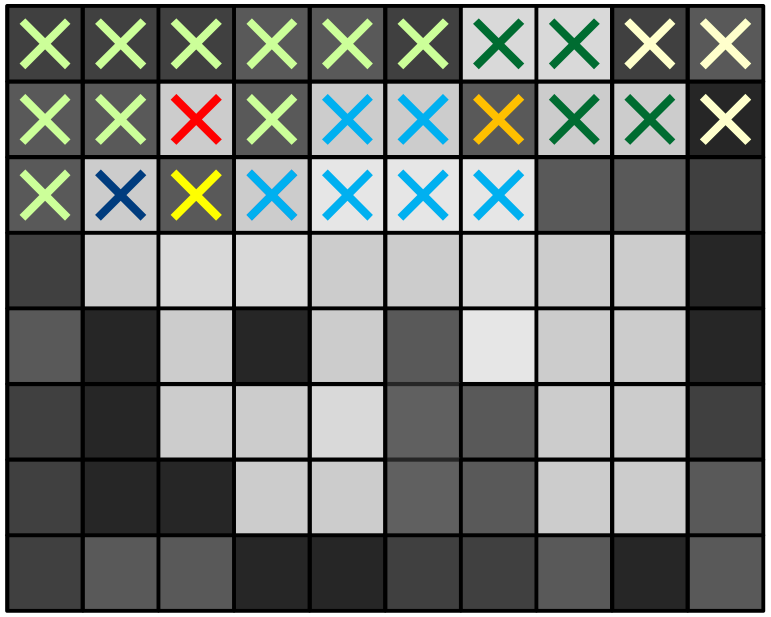


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

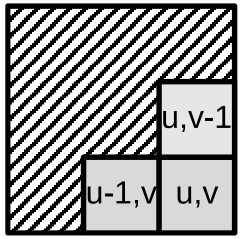


- Example

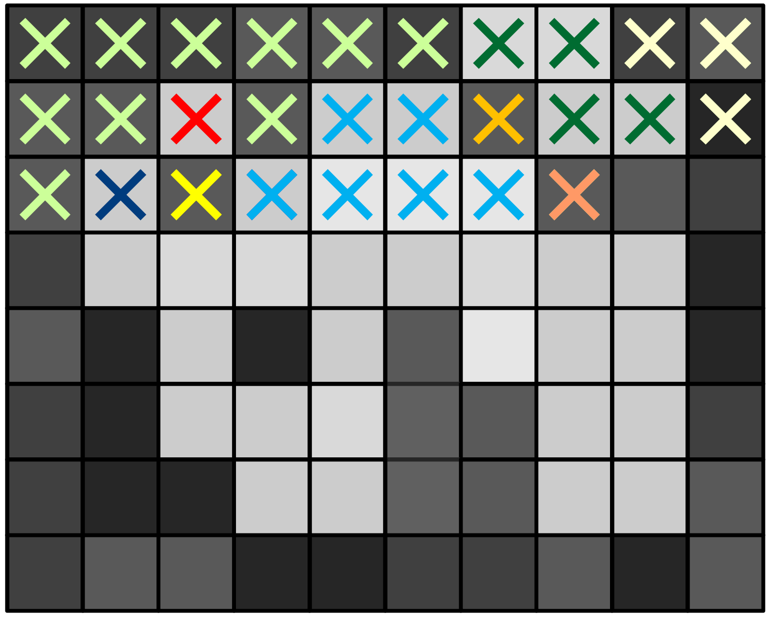


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

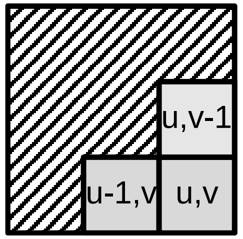


- Example

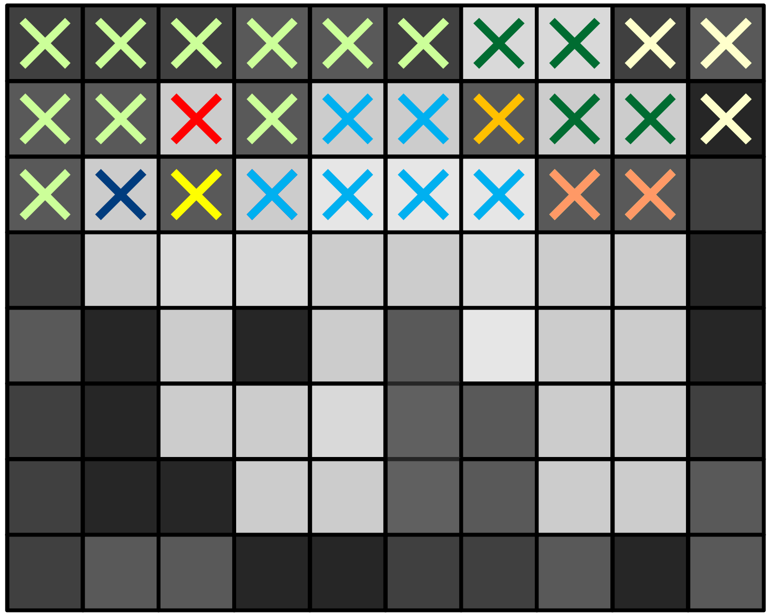


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

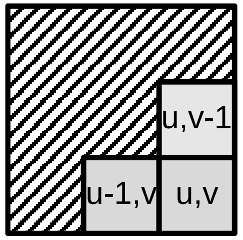


- Example

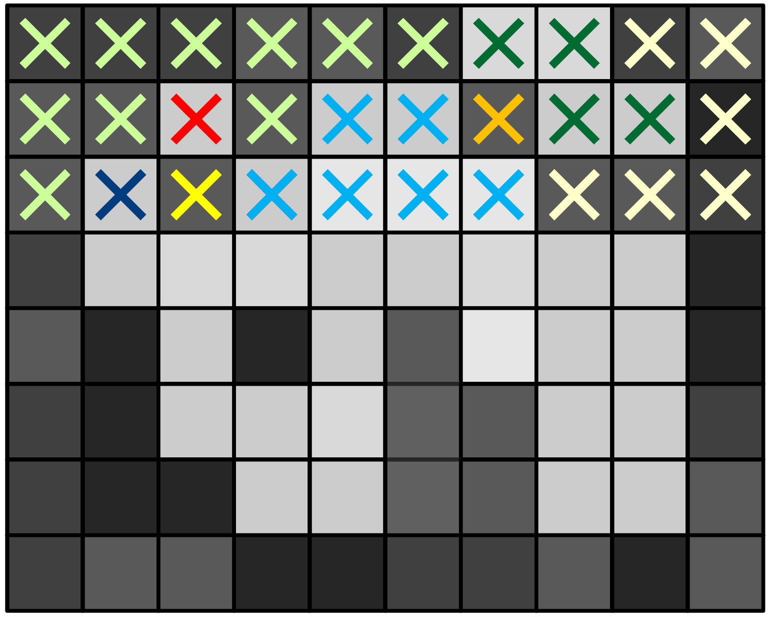


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

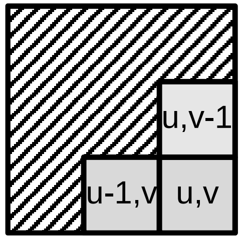


- Example

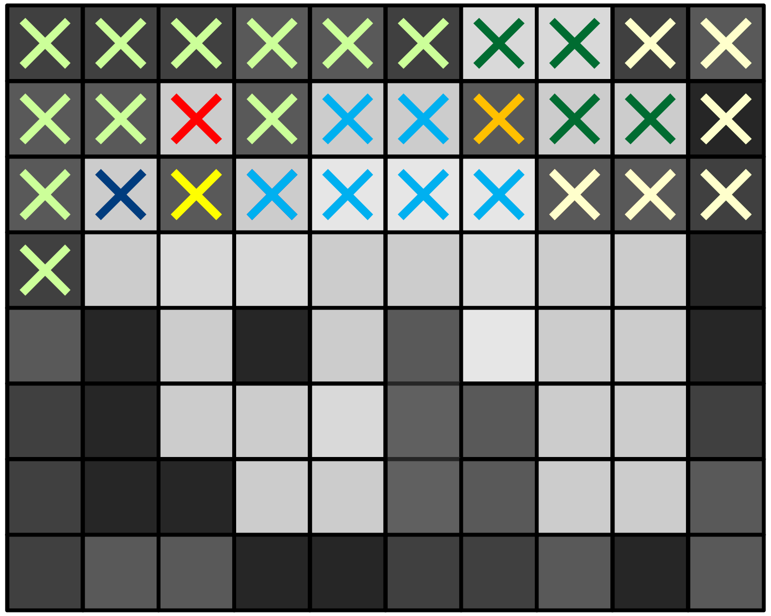


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment



- Example



Connected Components Labeling (CCL)

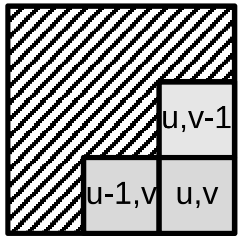
5. pixel colors at (u,v) and (u-1,v) are similar

   pixel colors at (u,v) and (u,v-1) are similar

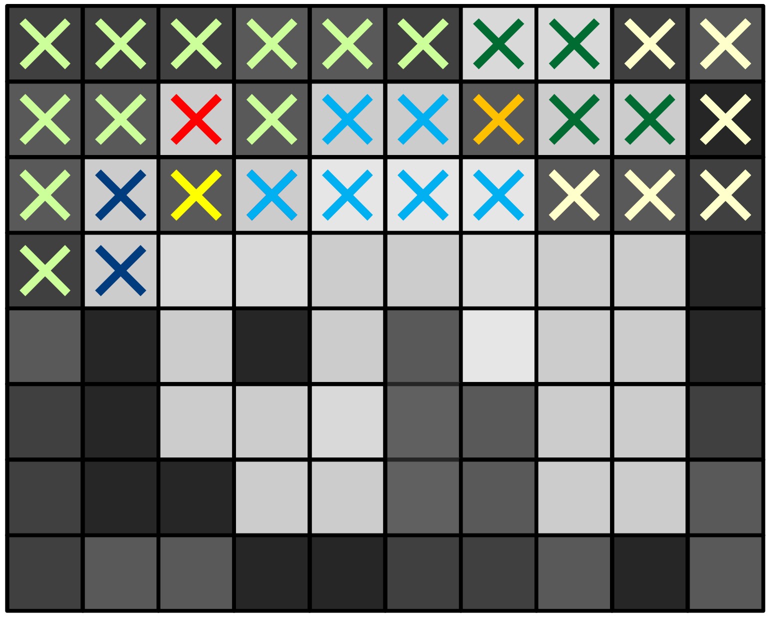
   pixels (u-1,v) and (u,v-1) do not belong to the same segment

   → pixel (u,v) belongs to the segments of both neighbors

   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

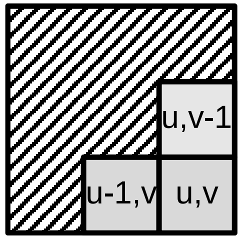


- Example

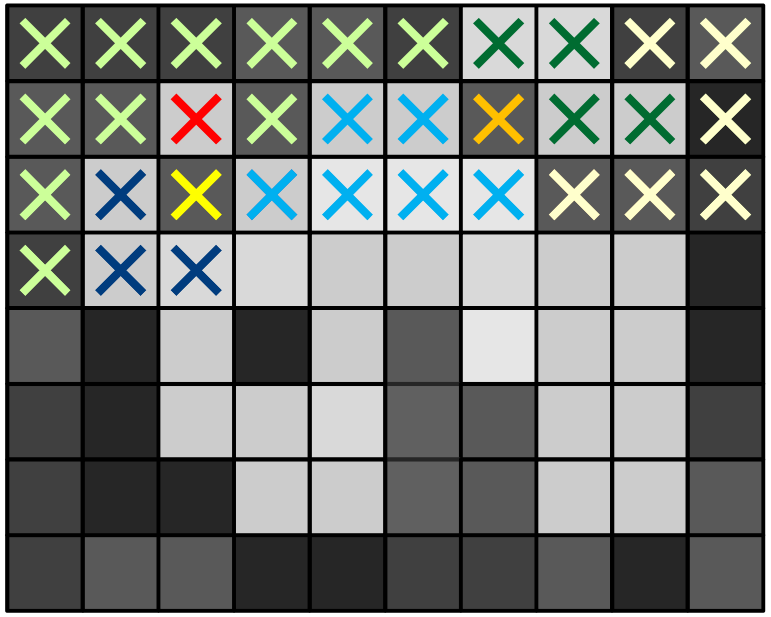


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

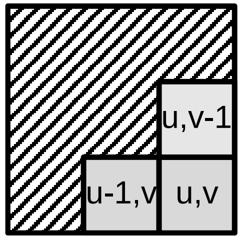


- Example

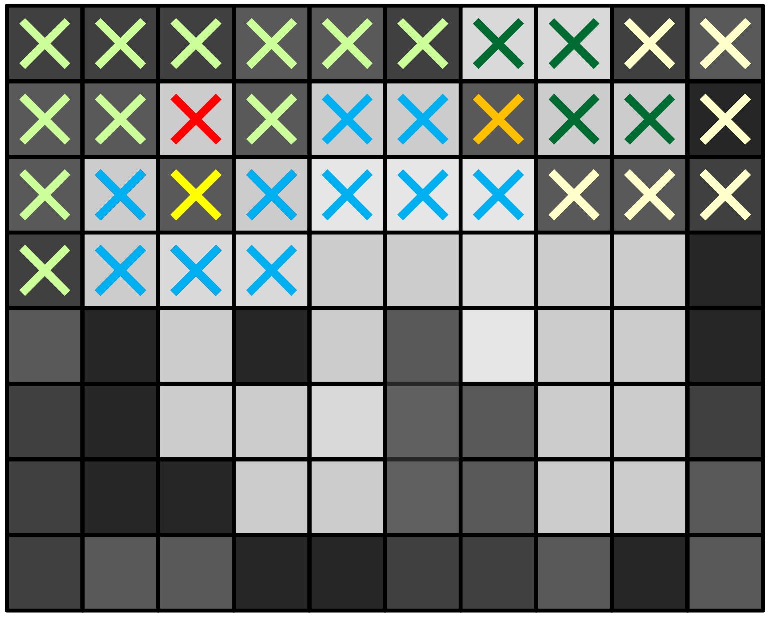


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

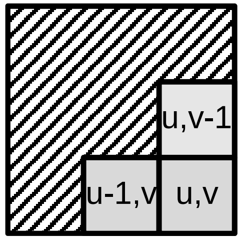


- Example

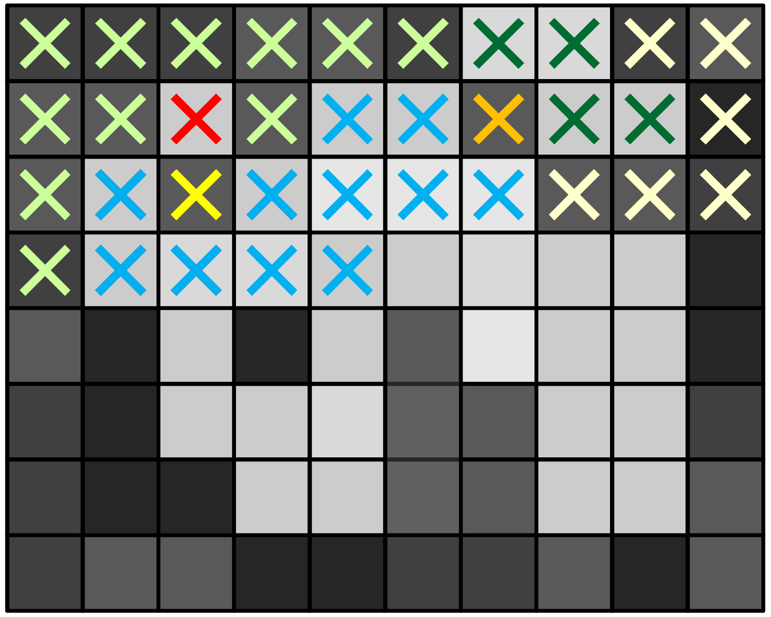


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

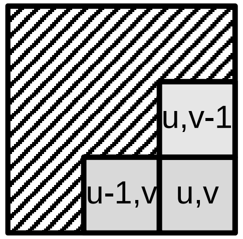


- Example

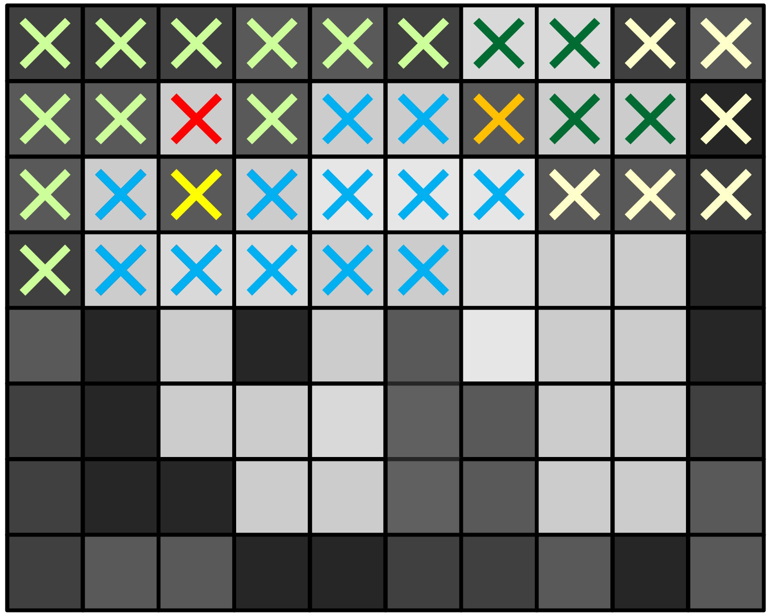


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment



- Example



Connected Components Labeling (CCL)

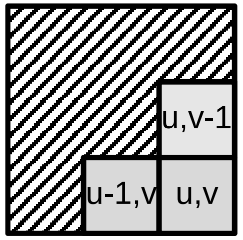
5. pixel colors at (u,v) and (u-1,v) are similar

   pixel colors at (u,v) and (u,v-1) are similar

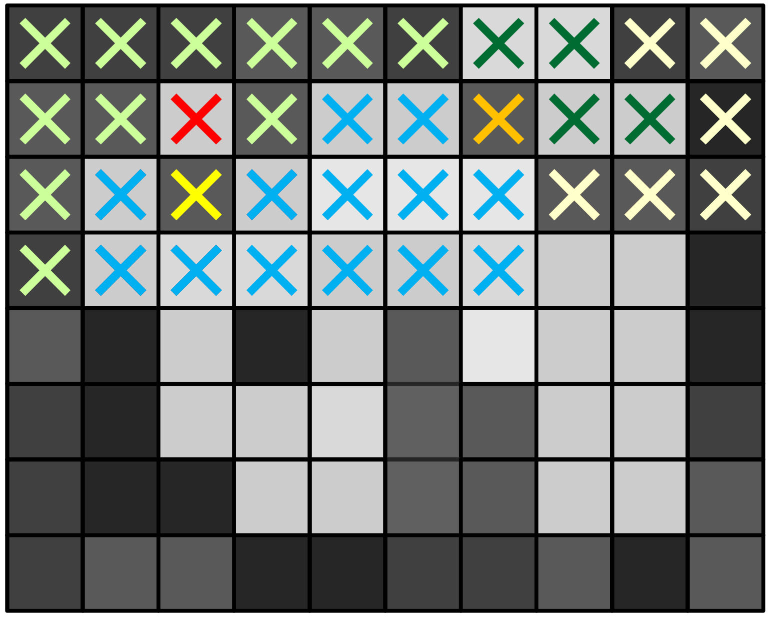
   pixels (u-1,v) and (u,v-1) do not belong to the same segment

   → pixel (u,v) belongs to the segments of both neighbors

   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

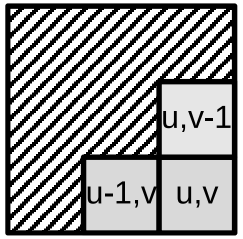


- Example

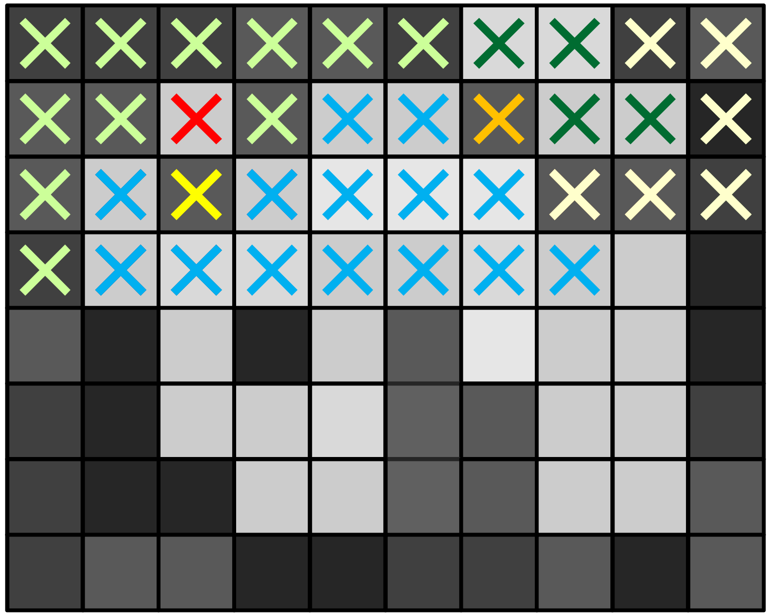


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

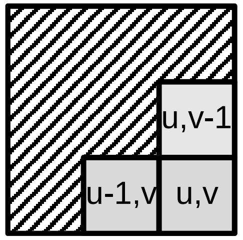


- Example

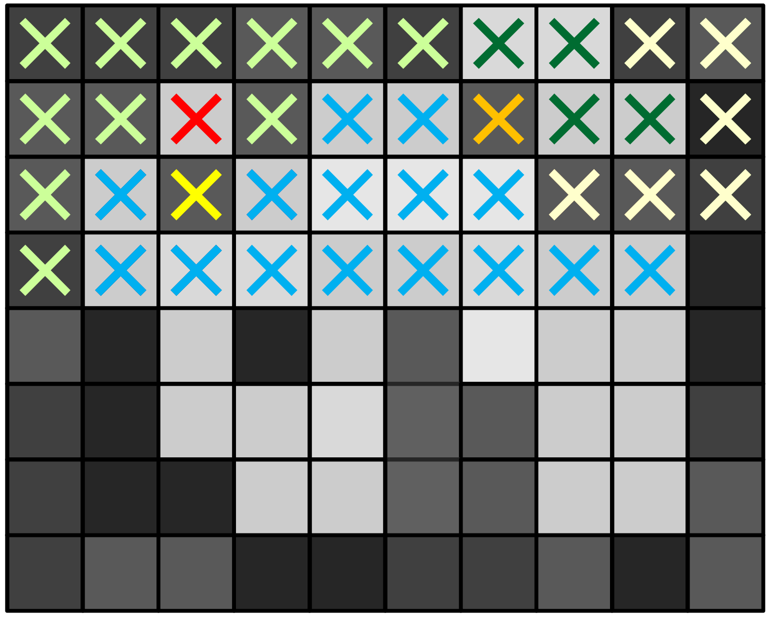


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

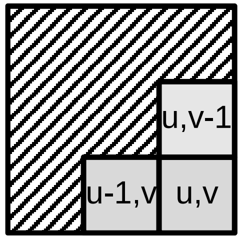


- Example

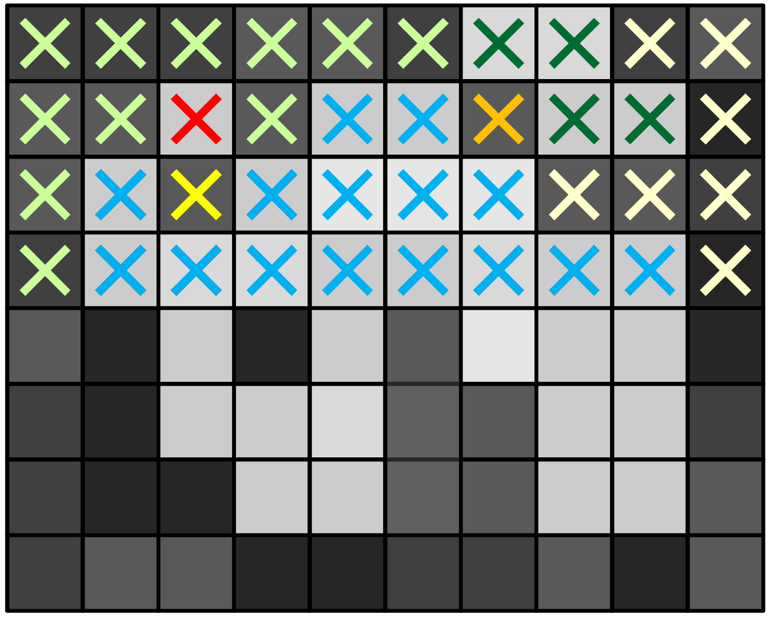


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

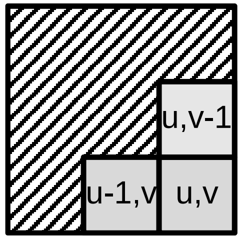


- Example

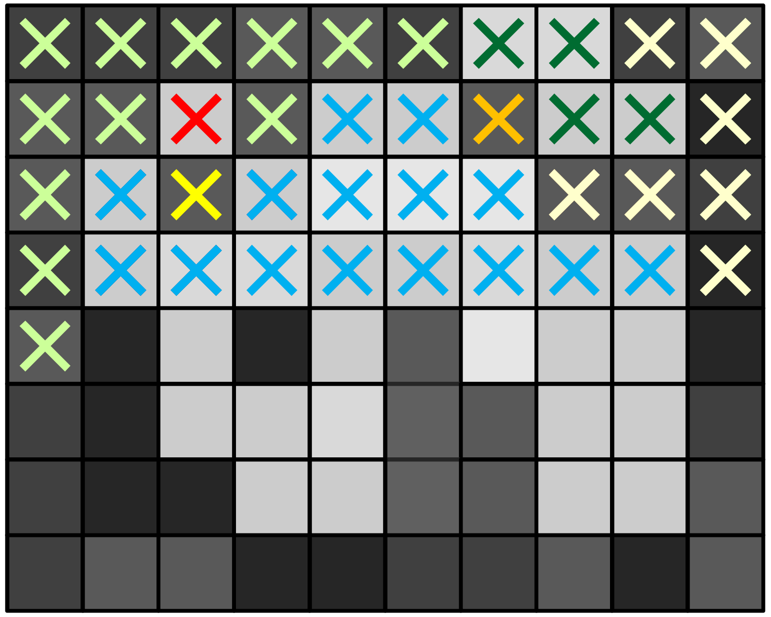


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

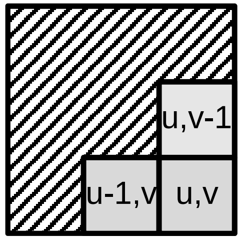


- Example

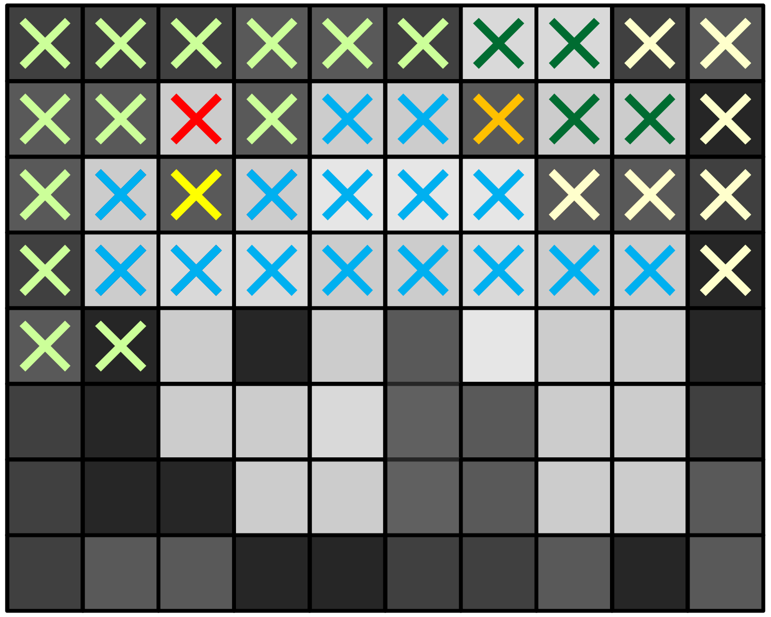


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

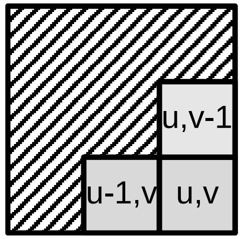


- Example

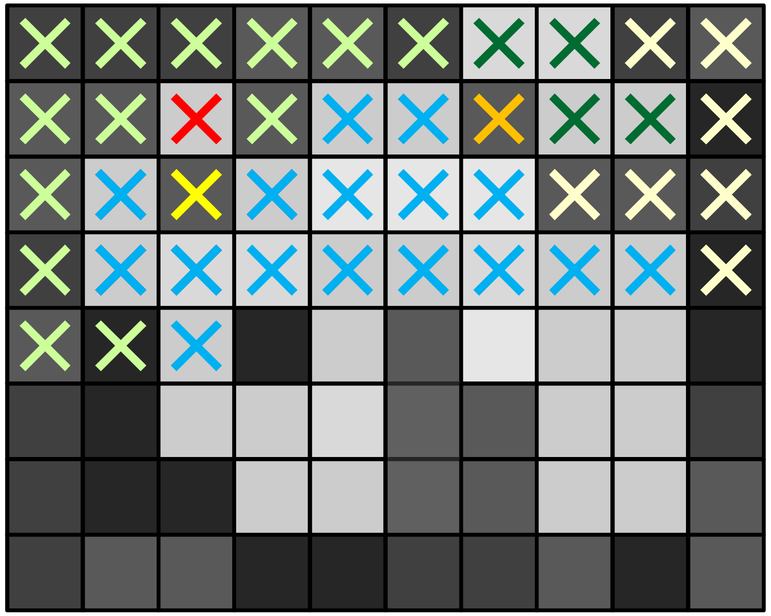


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

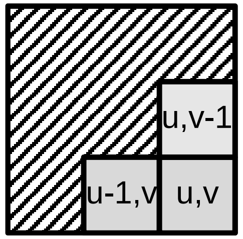


- Example

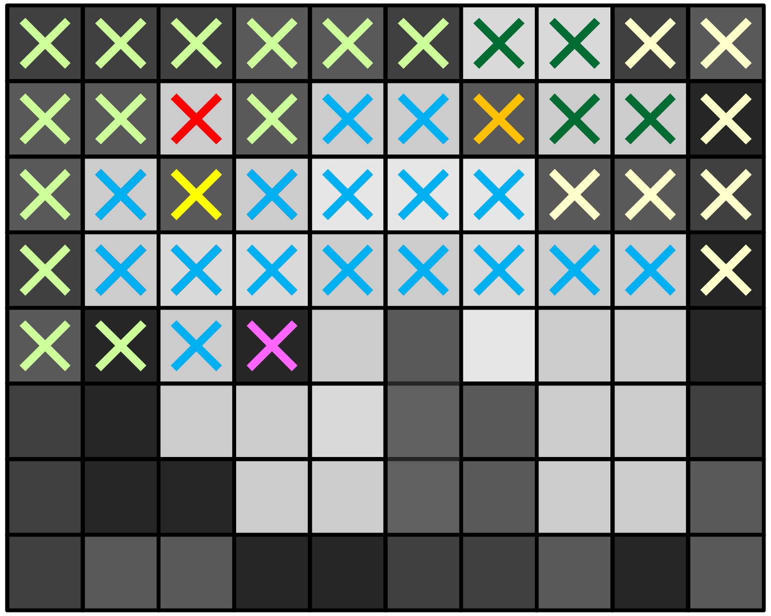


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

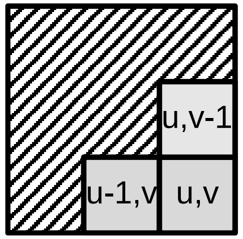


- Example

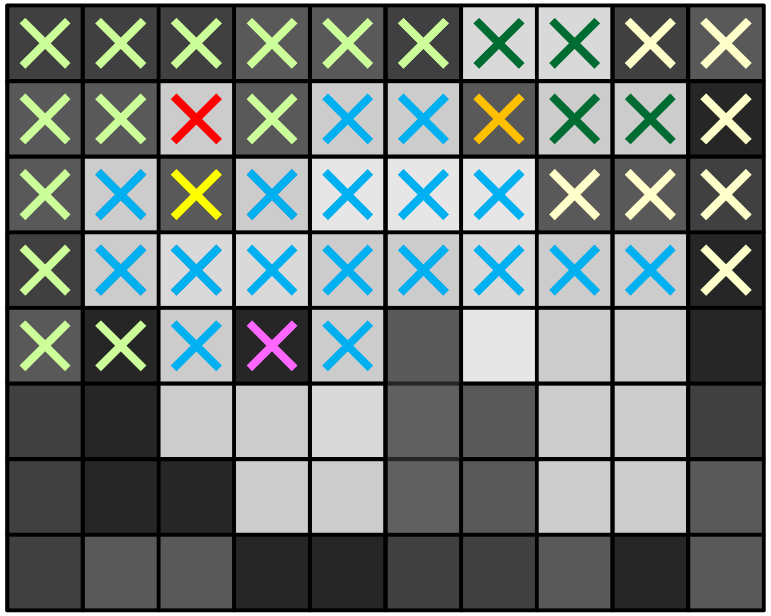


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

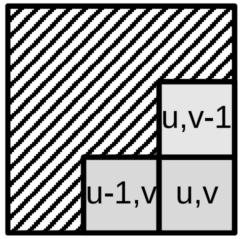


- Example

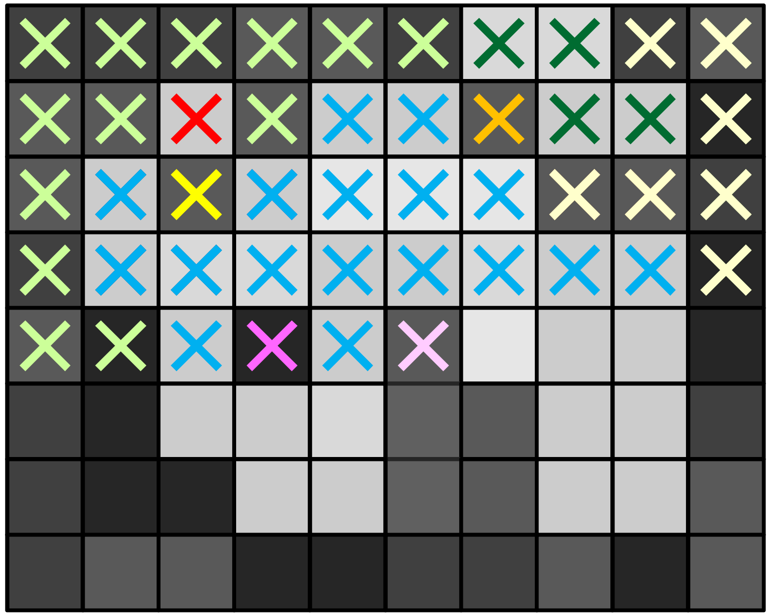


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

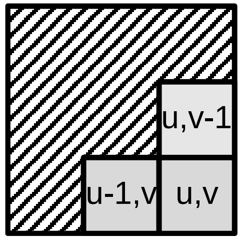


- Example

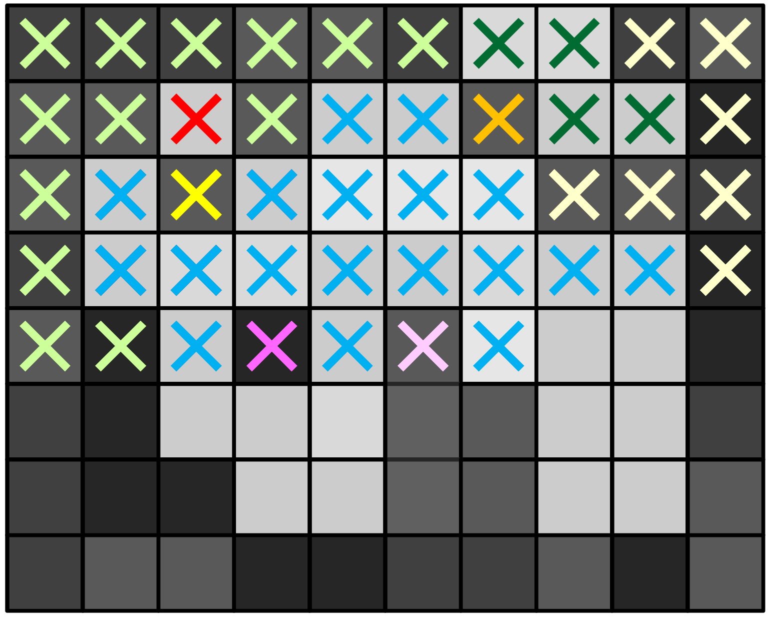


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

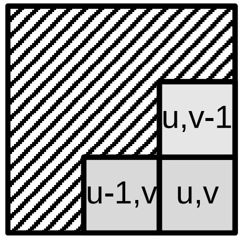


- Example

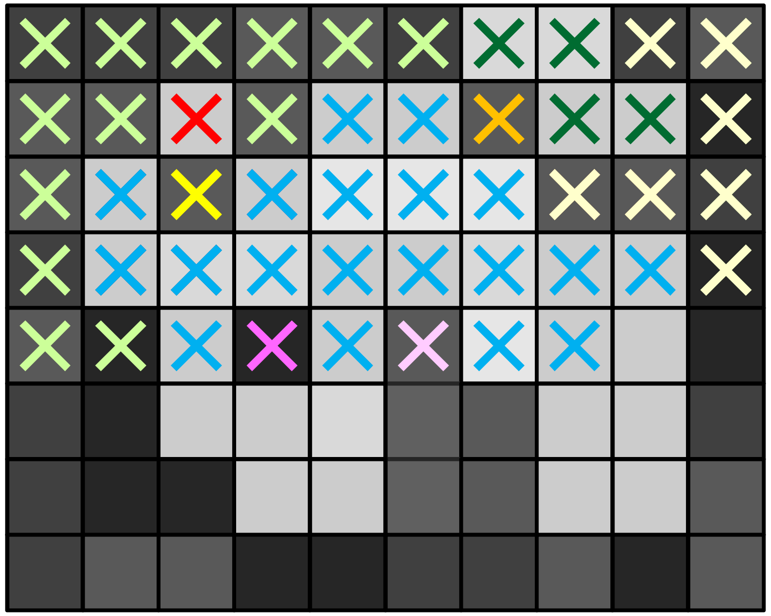


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

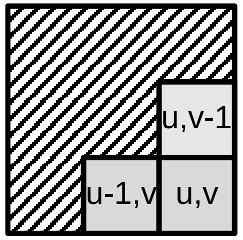


- Example

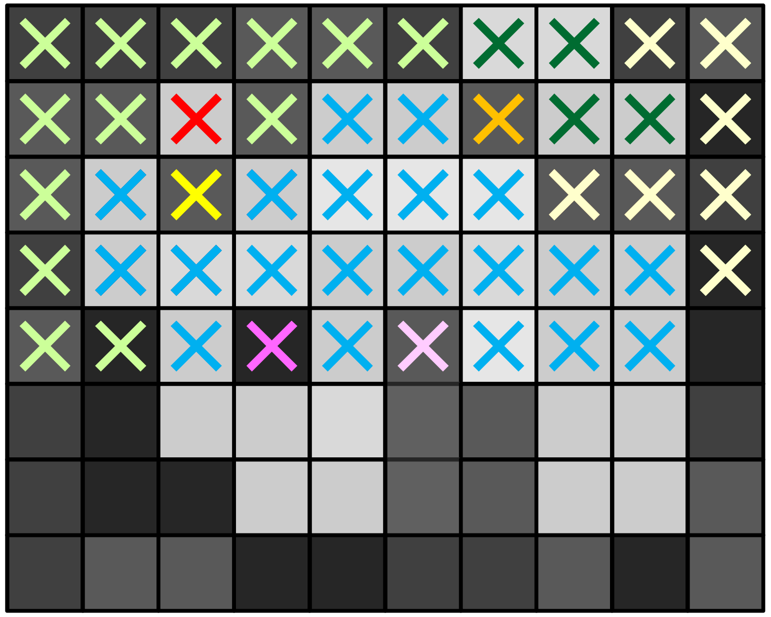


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

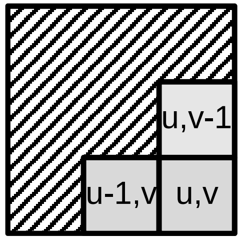


- Example

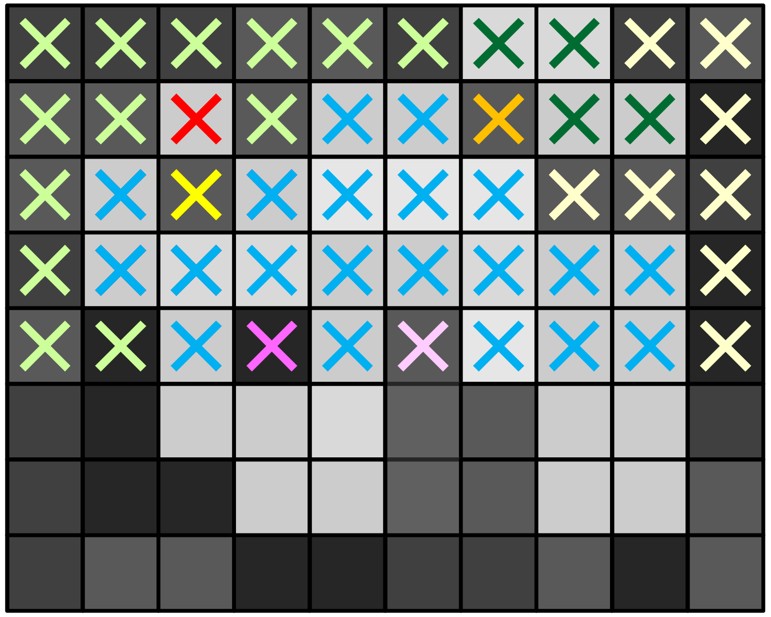


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

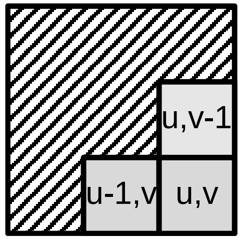


- Example

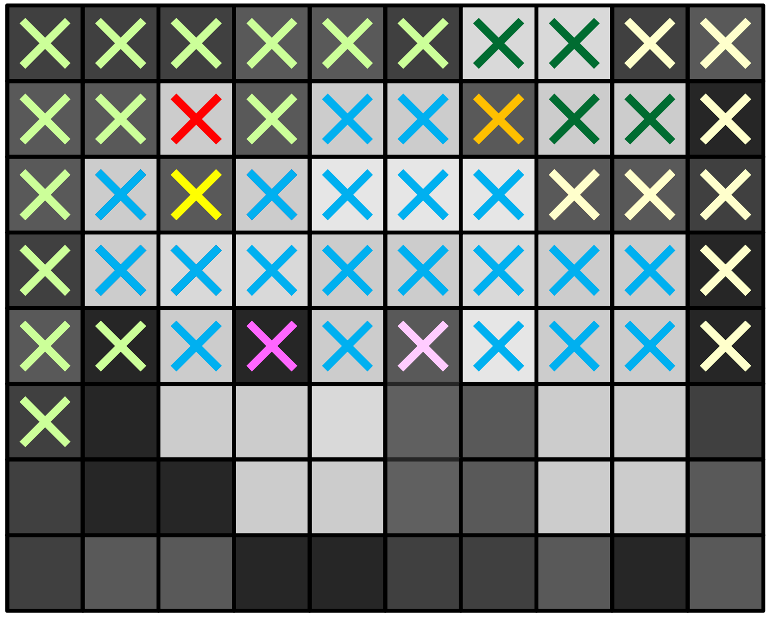


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

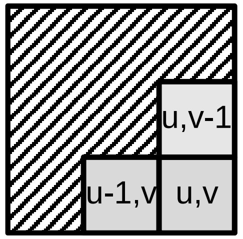


- Example

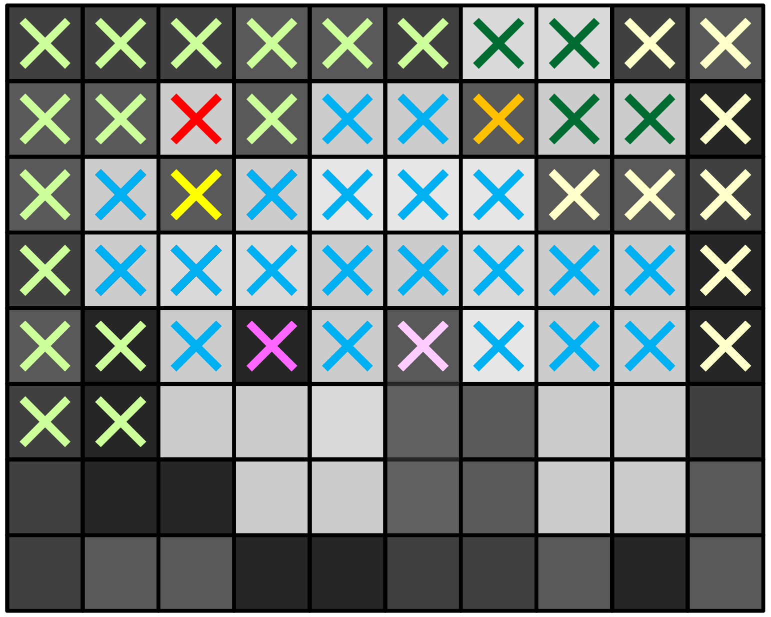


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

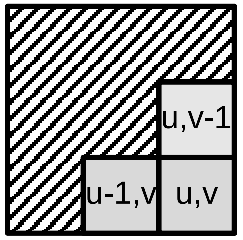


- Example

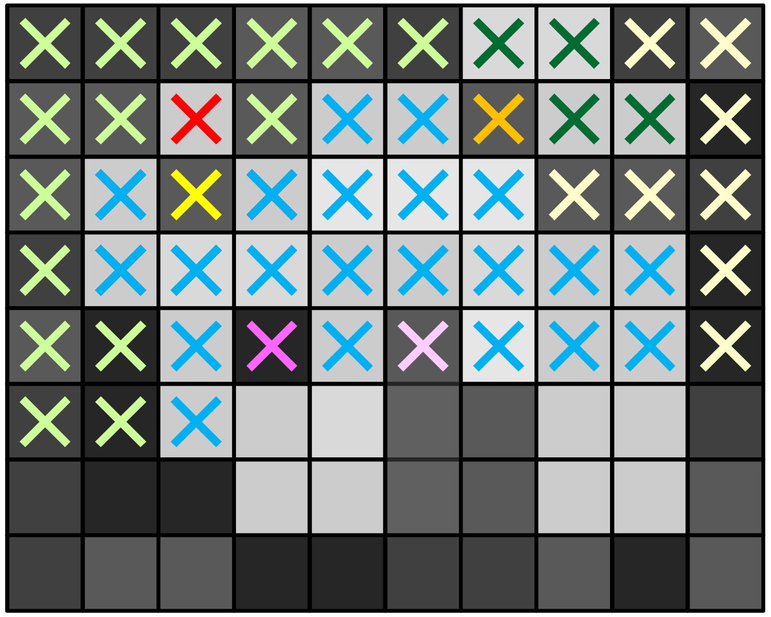


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

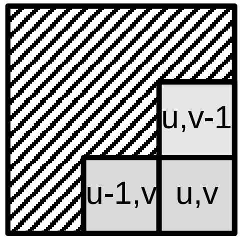


- Example

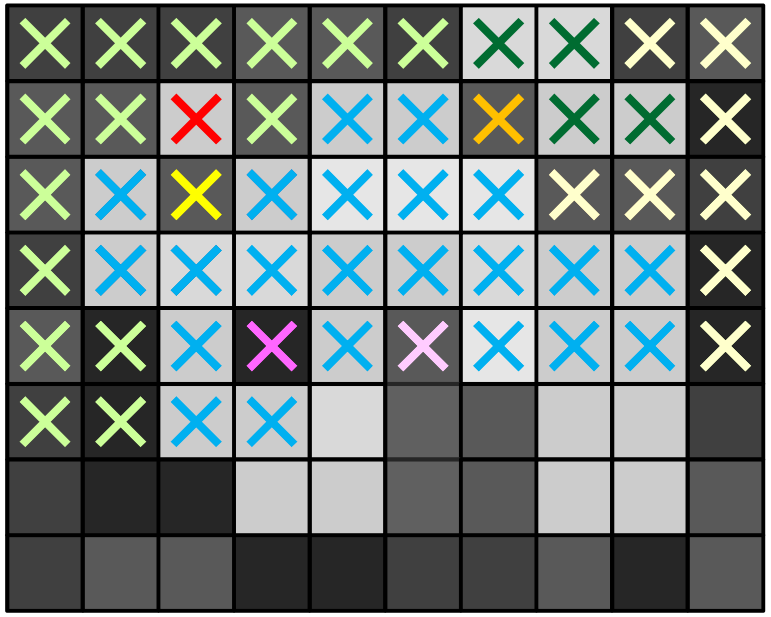


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment



- Example



Connected Components Labeling (CCL)

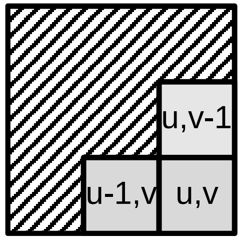
5. pixel colors at (u,v) and (u-1,v) are similar

   pixel colors at (u,v) and (u,v-1) are similar

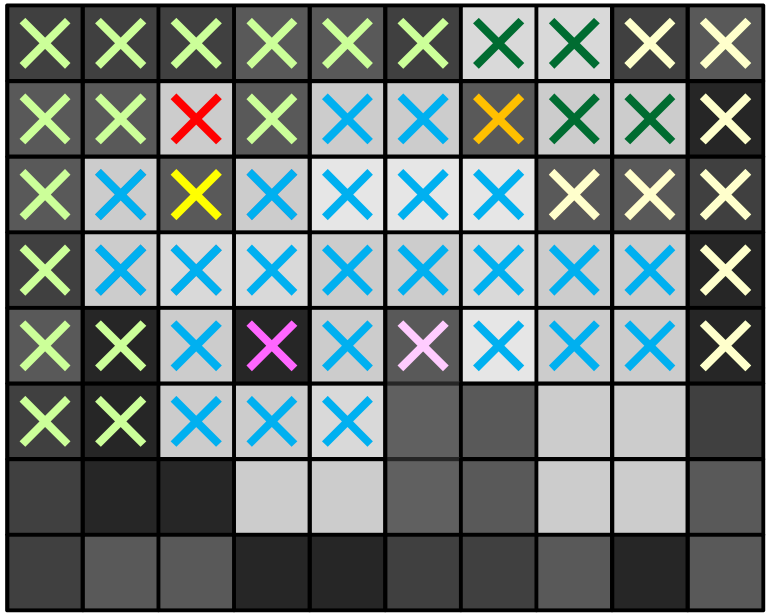
   pixels (u-1,v) and (u,v-1) do not belong to the same segment

   → pixel (u,v) belongs to the segments of both neighbors

   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

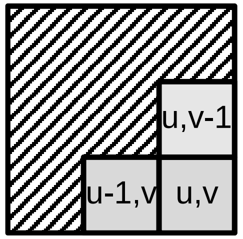


- Example

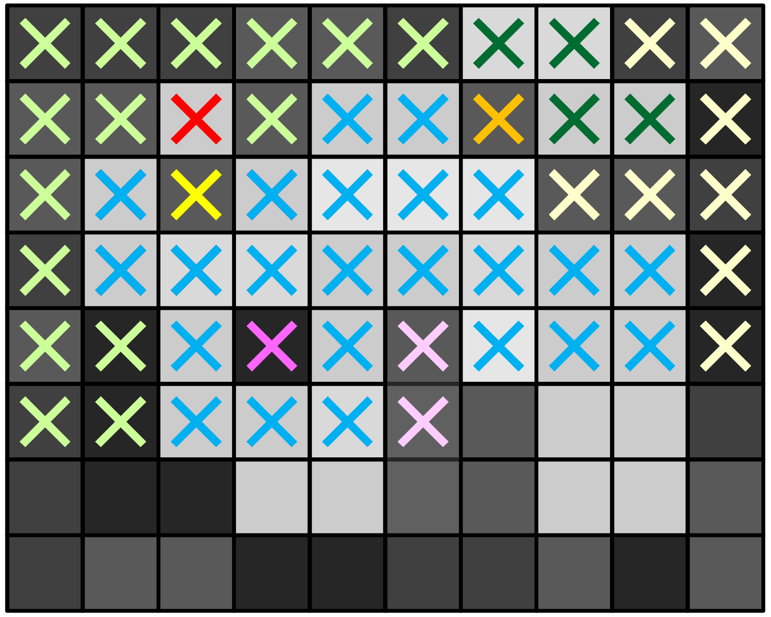


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

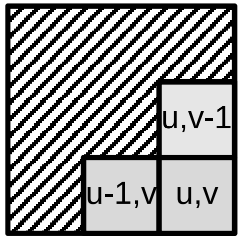


- Example

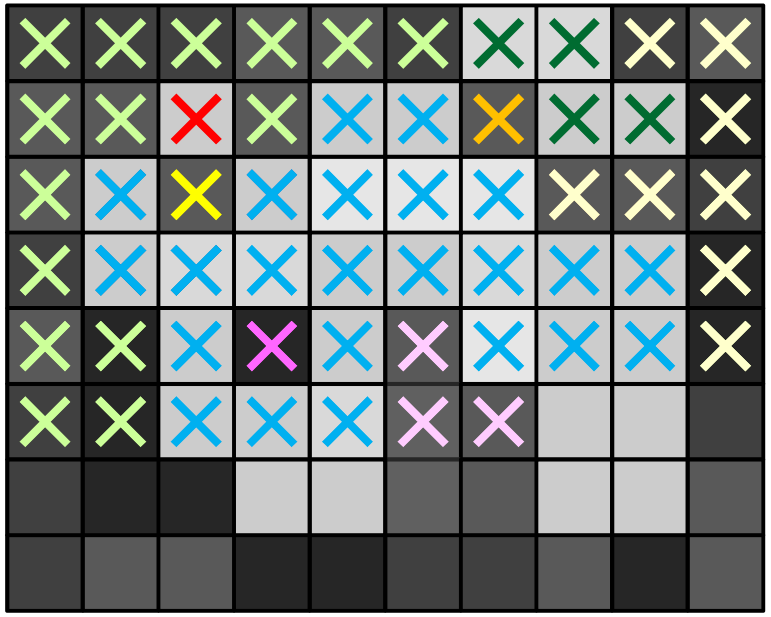


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

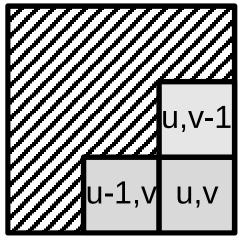


- Example

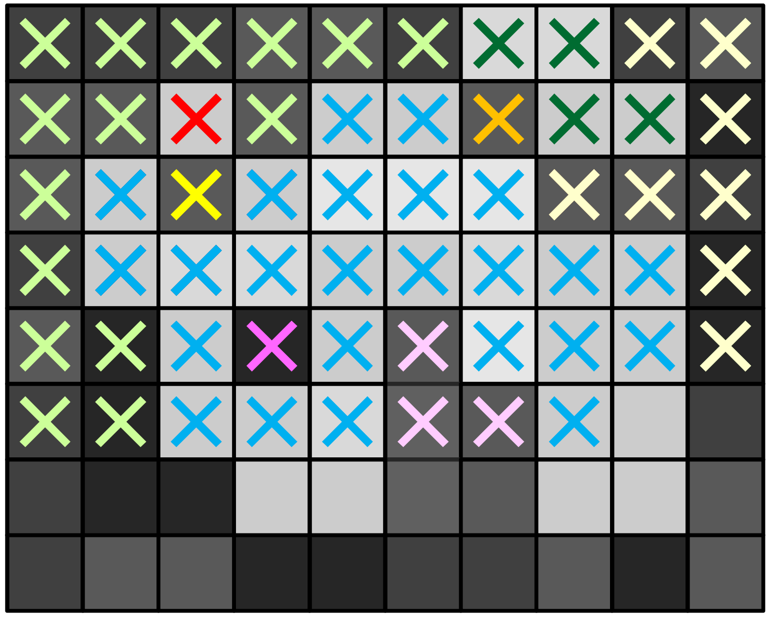


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

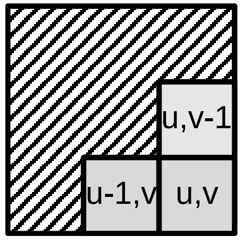


- Example

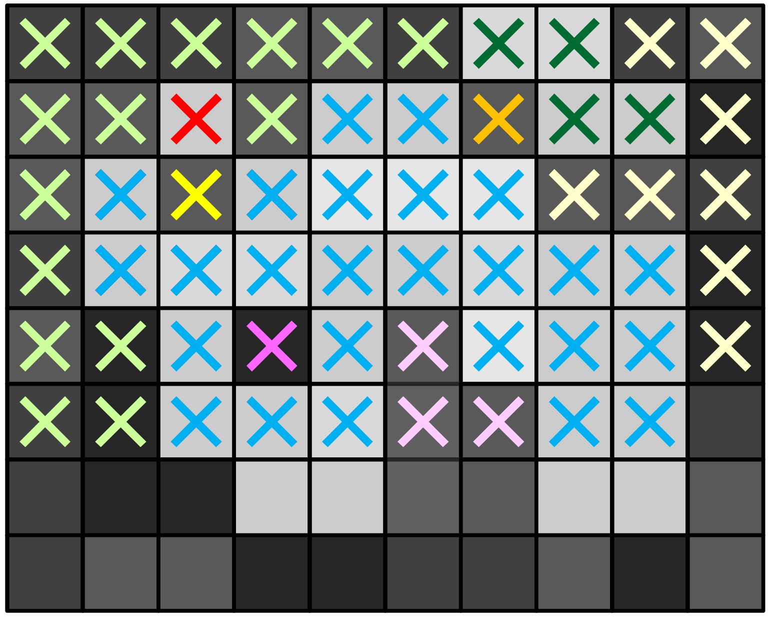


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment



- Example



Connected Components Labeling (CCL)

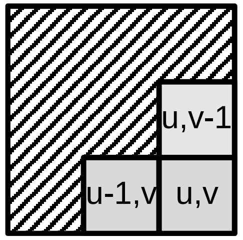
5. pixel colors at (u,v) and (u-1,v) are similar

   pixel colors at (u,v) and (u,v-1) are similar

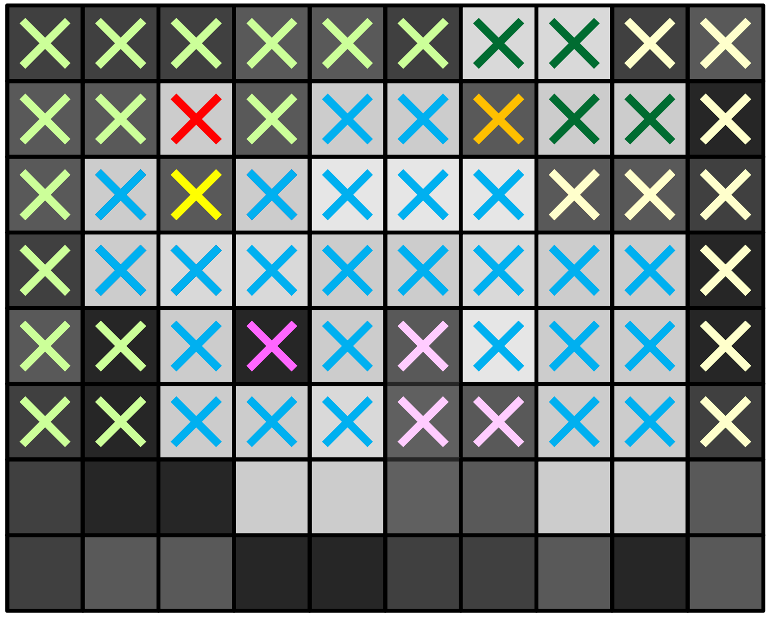
   pixels (u-1,v) and (u,v-1) do not belong to the same segment

   → pixel (u,v) belongs to the segments of both neighbors

   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

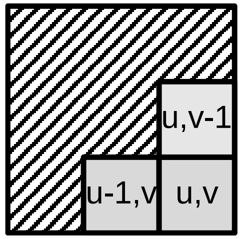


- Example

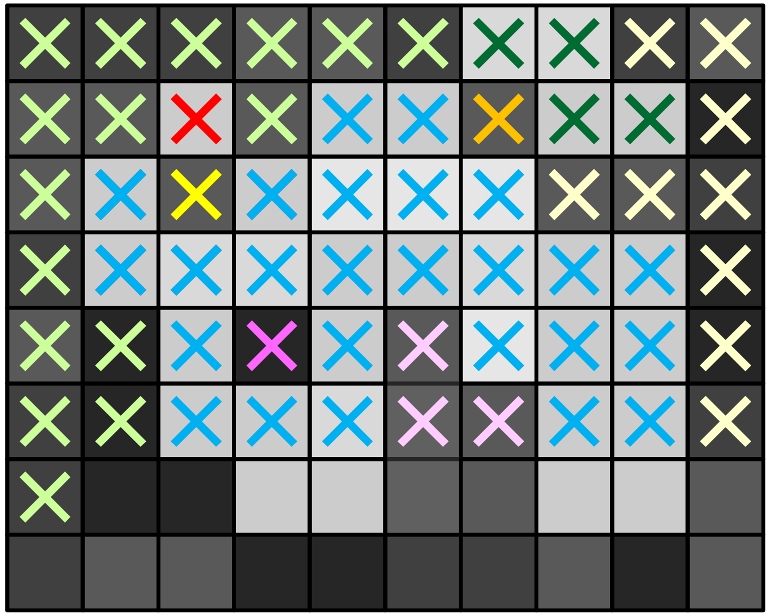


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

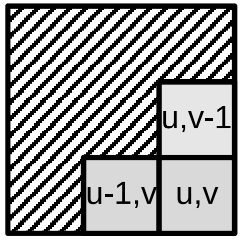


- Example

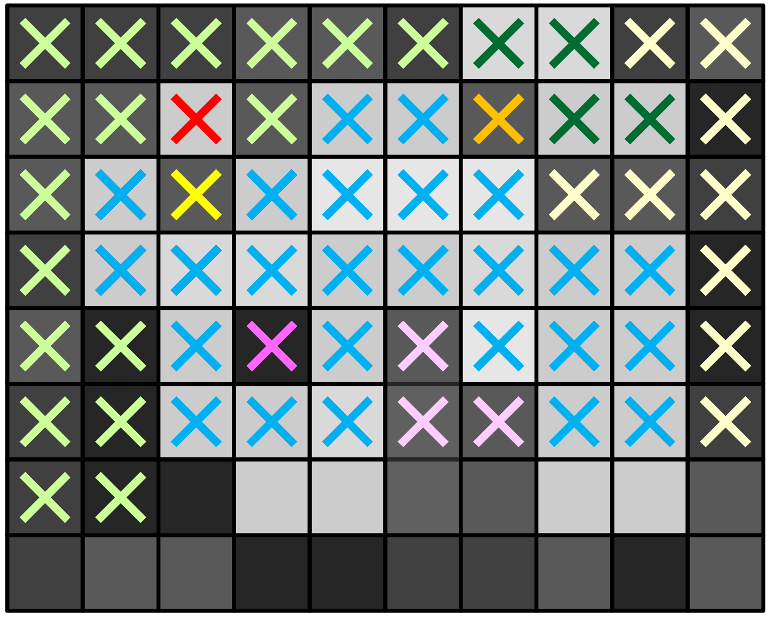


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

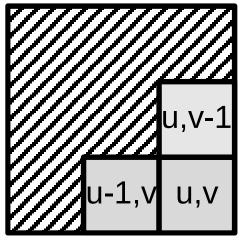


- Example

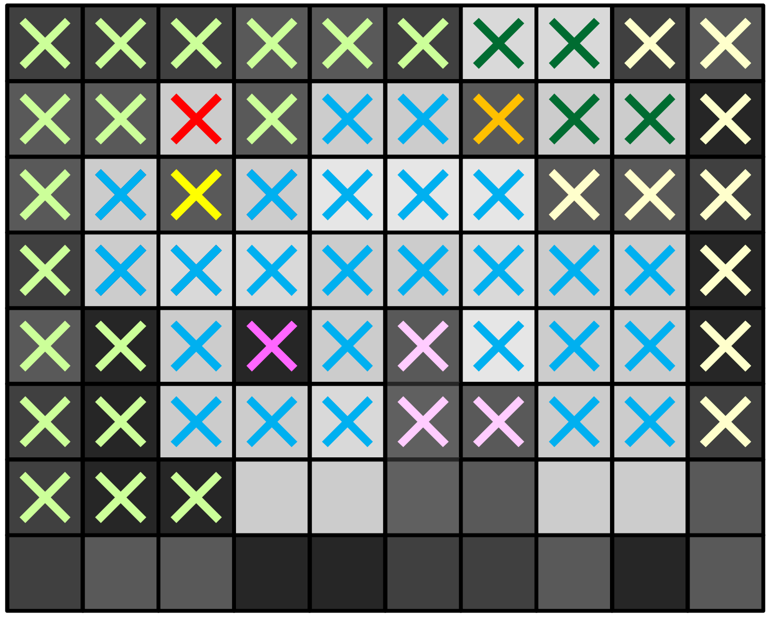


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

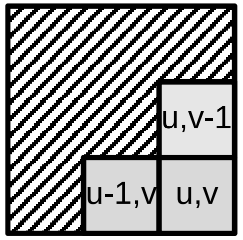


- Example

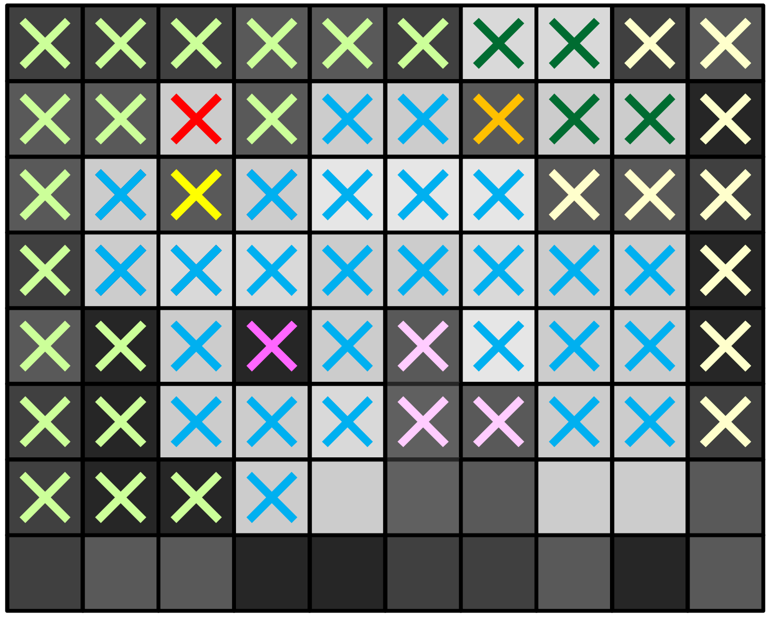


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

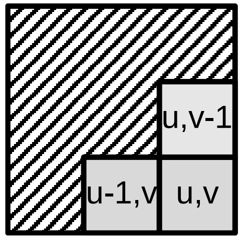


- Example

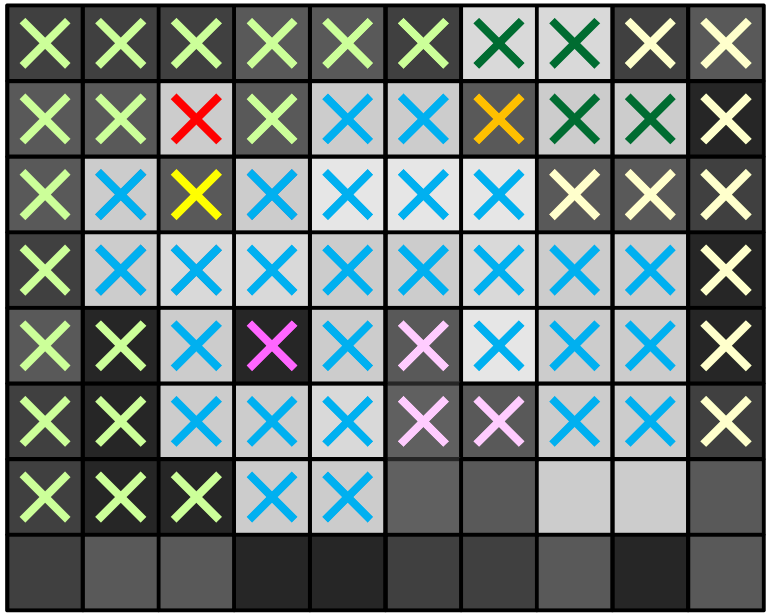


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

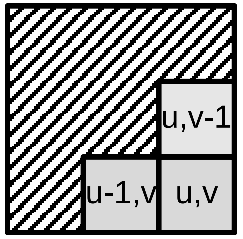


- Example

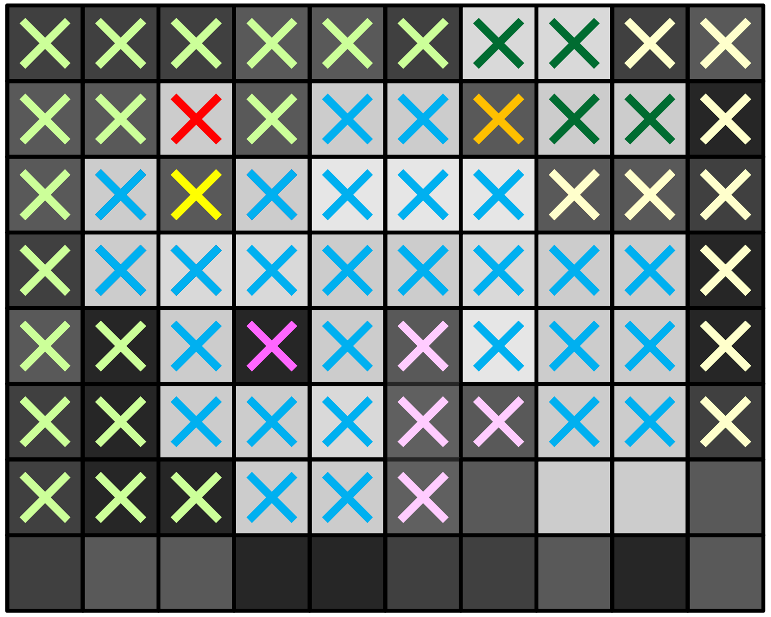


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

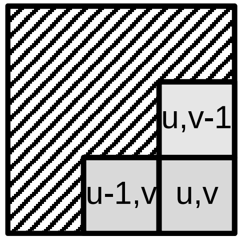


- Example

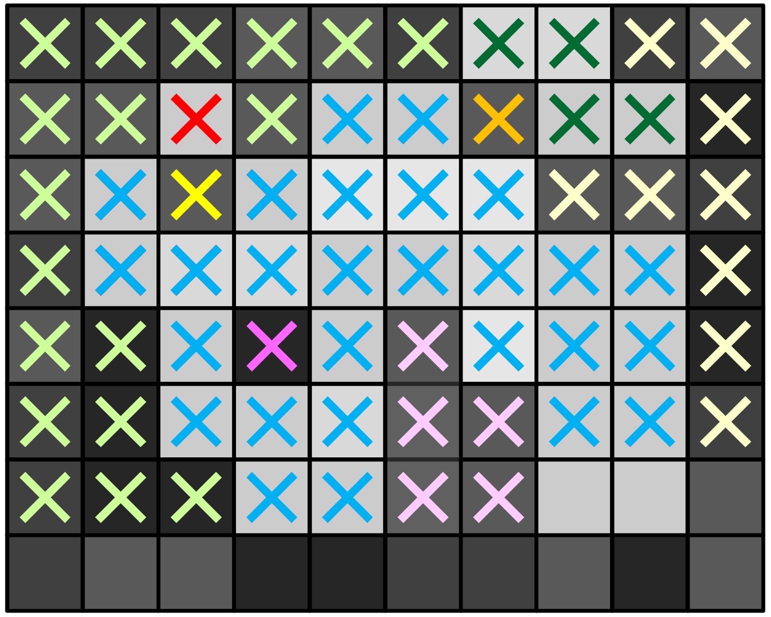


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

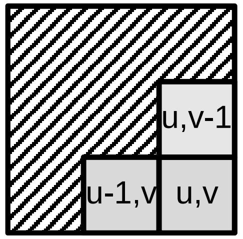


- Example

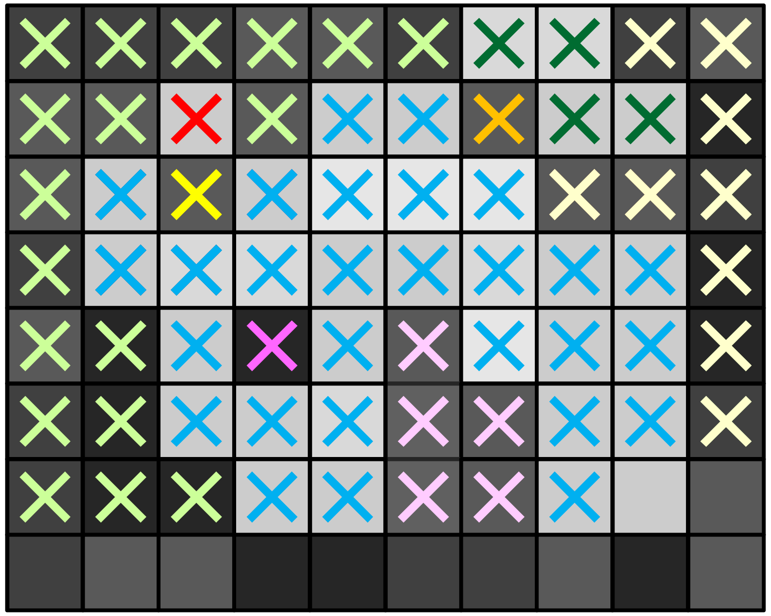


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

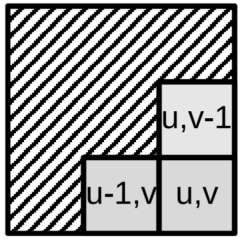


- Example

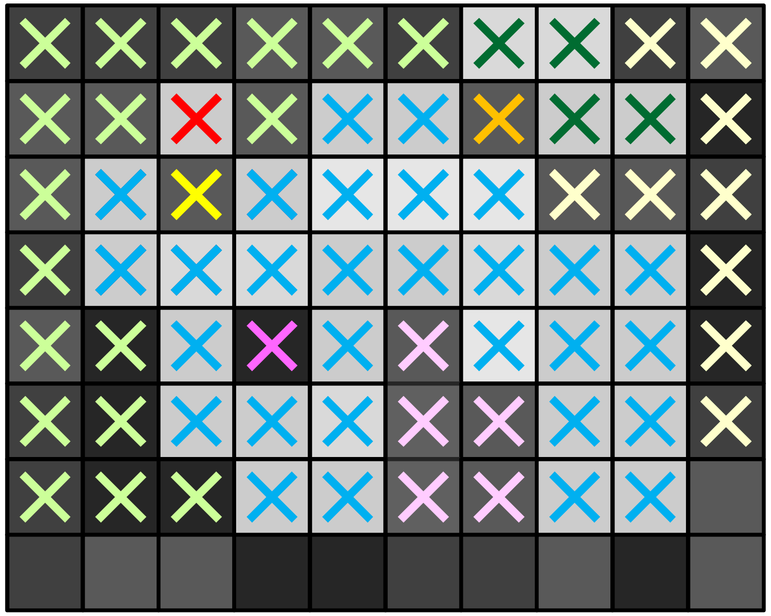


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

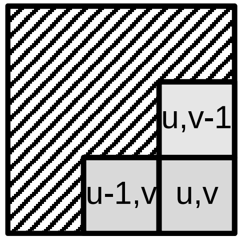


- Example

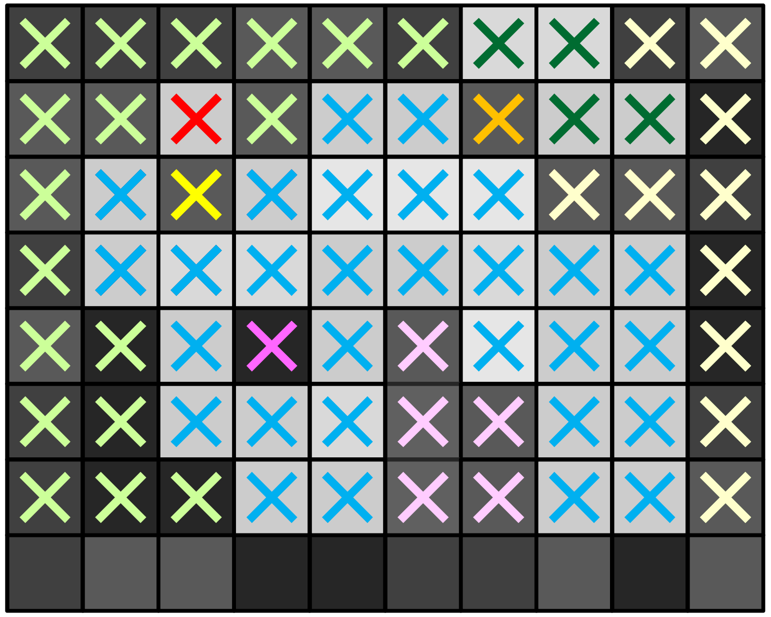


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

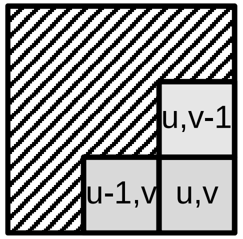


- Example

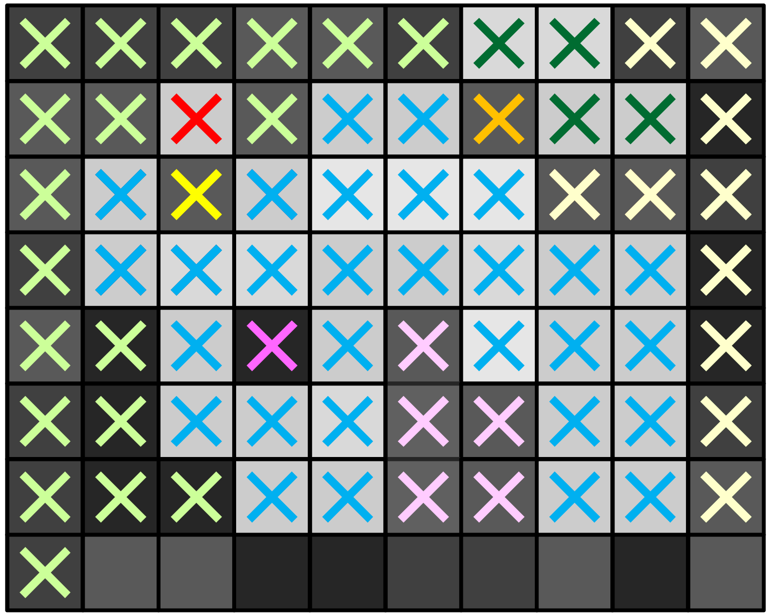


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

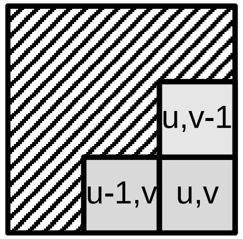


- Example

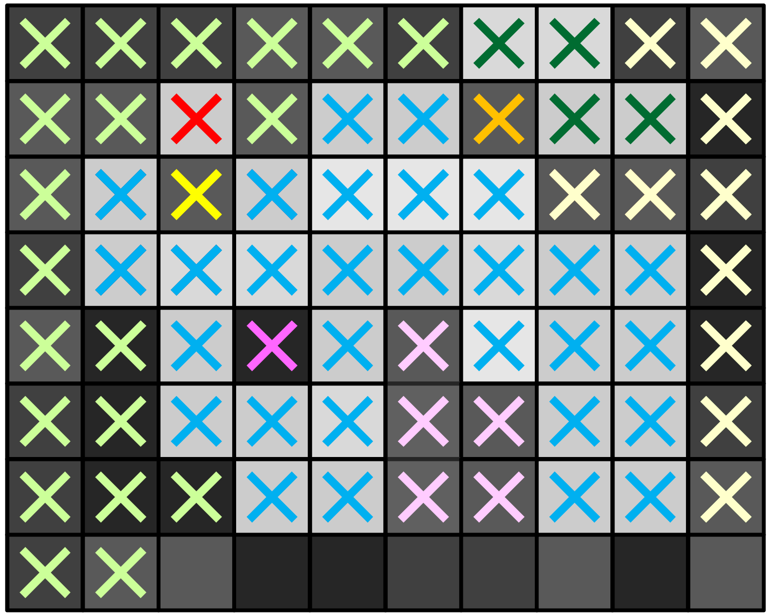


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

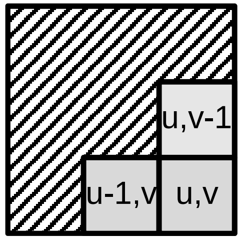


- Example

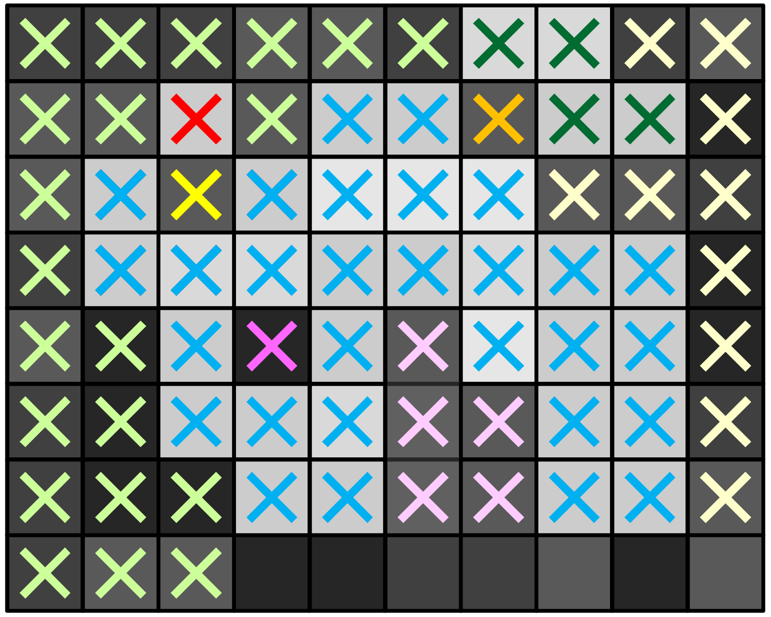


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

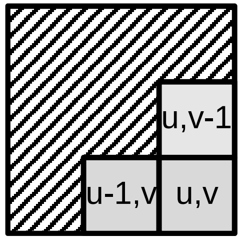


- Example

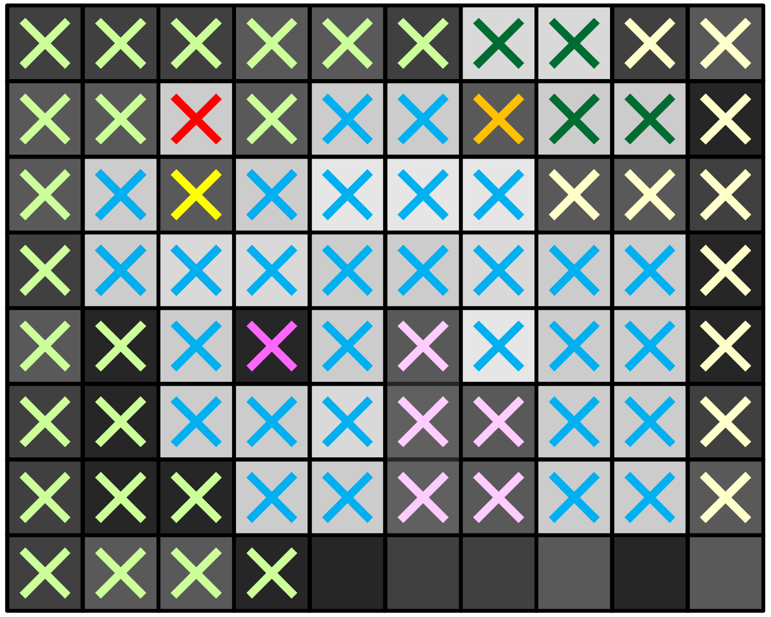


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

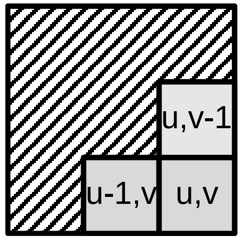


- Example

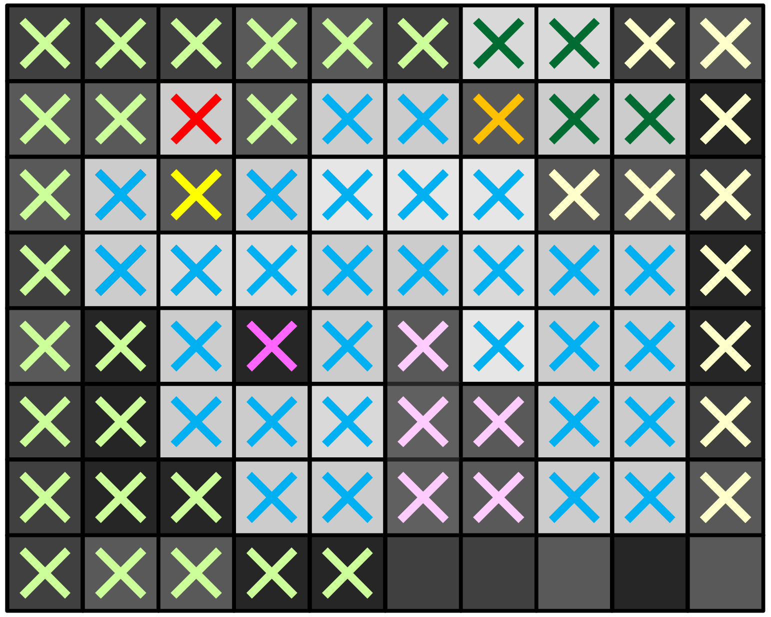


Connected Components Labeling (CCL)

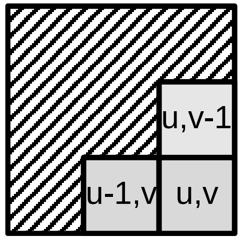
5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment



- Example

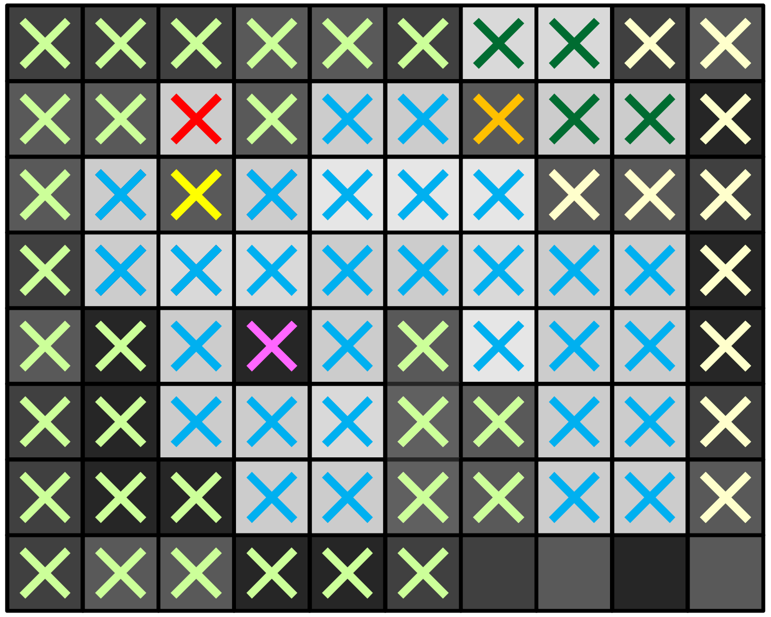


Connected Components Labeling (CCL)



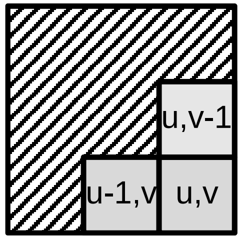
5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

- Example

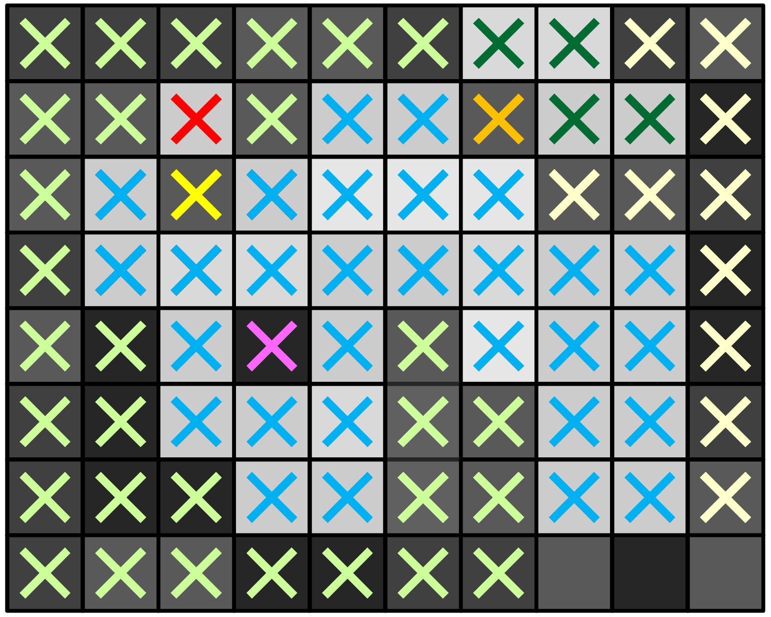


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

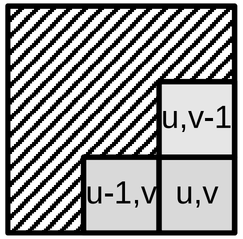


- Example

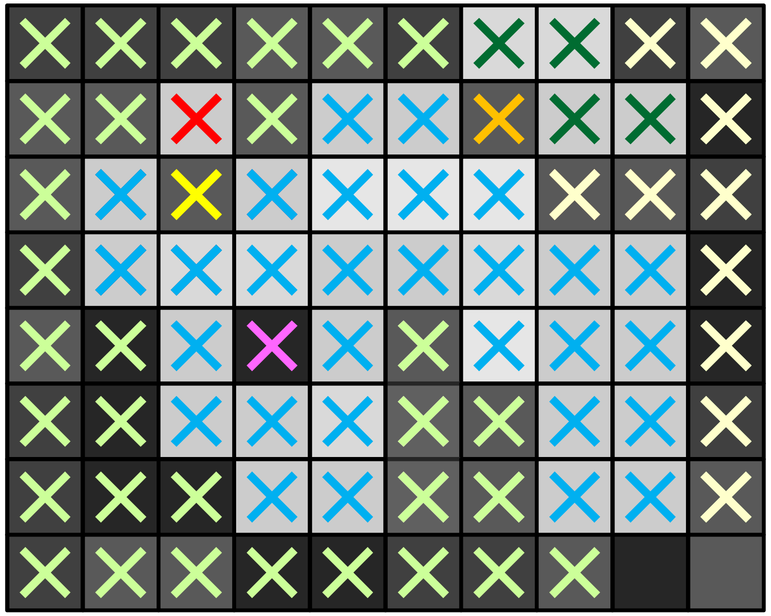


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

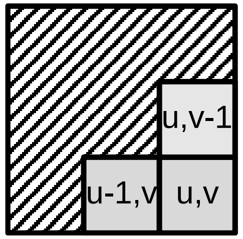


- Example

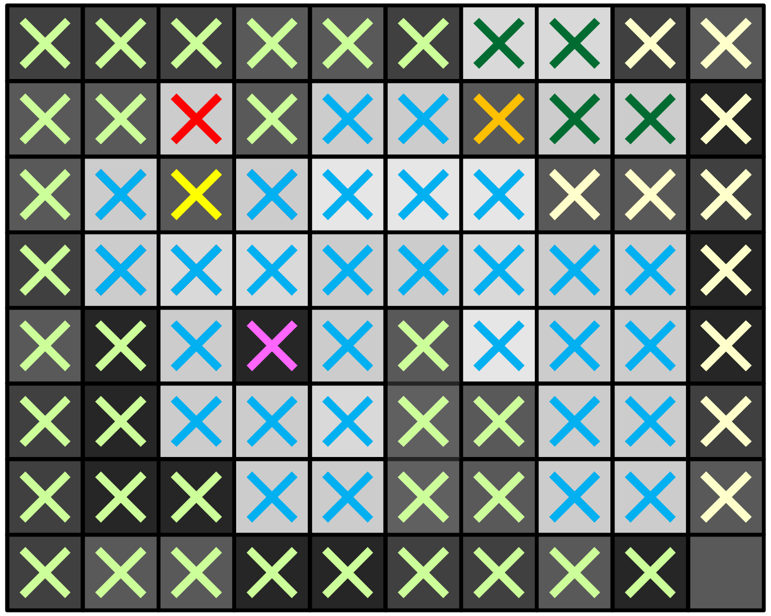


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment

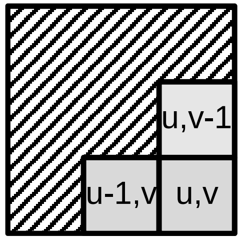


- Example

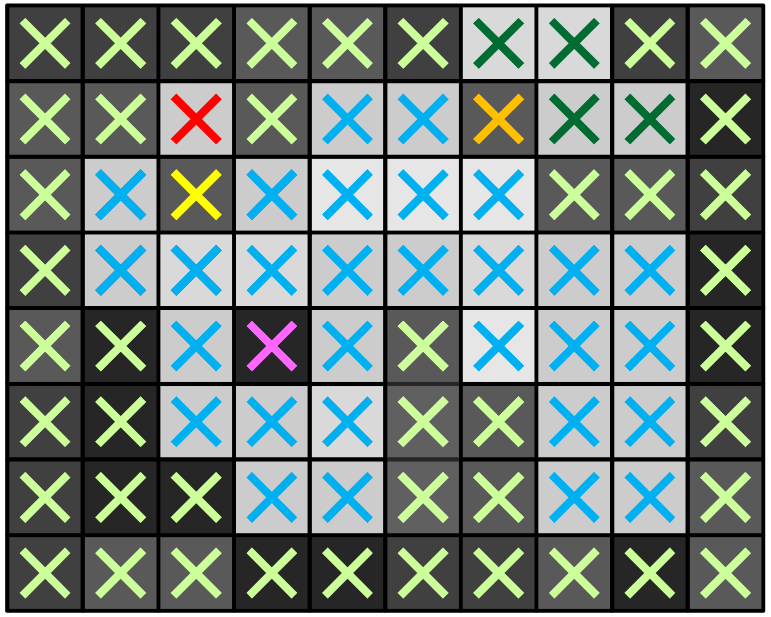


Connected Components Labeling (CCL)

5. pixel colors at (u,v) and (u-1,v) are similar
   pixel colors at (u,v) and (u,v-1) are similar
   pixels (u-1,v) and (u,v-1) do not belong to the same segment
   → pixel (u,v) belongs to the segments of both neighbors
   → we merge the two neighboring segments and assign pixel (u,v) to the merged segment



- Example

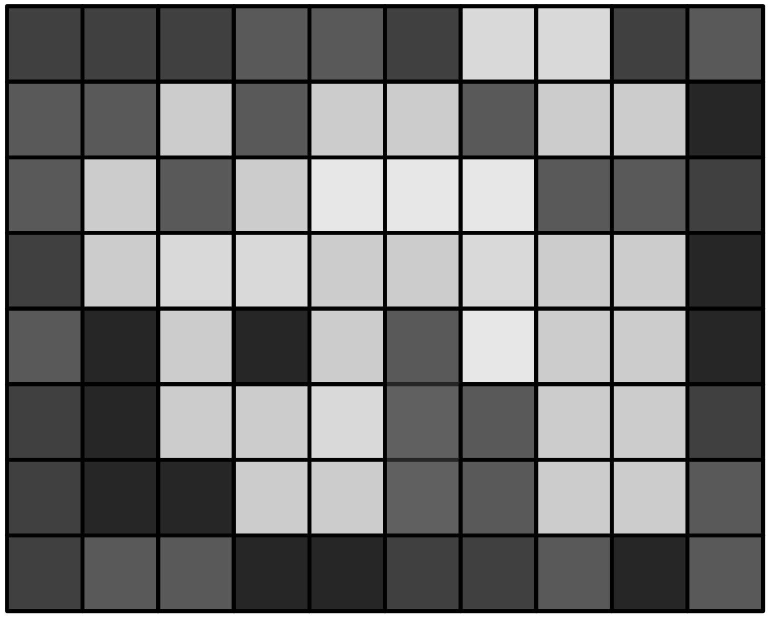


Slide 23

k-means

- how can we find color clusters?
- if we know the number of clusters
  → k-means algorithm

1. initialize *k* prototype colors $c_1, c_2, \dots, c_k$ randomly (e.g. by randomly picking pixels from image)
2. assign each pixel to the prototype color that is most similar
3. recalculate prototype colors by averaging over colors of pixel which have been assigned in step 2
4. repeat steps 2 and 3 until convergence (i.e. the assignments in step 2 do not change any more)



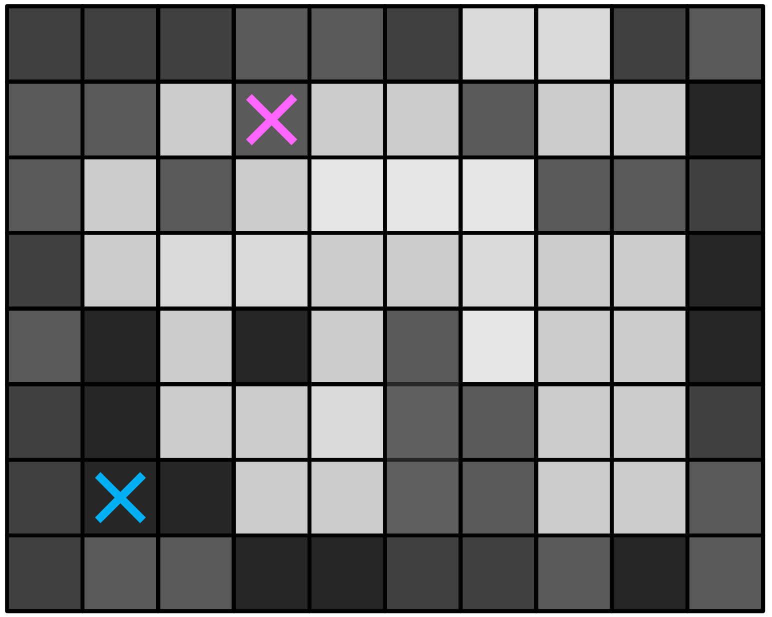
example: *k=2*

k-means

– how can we find color clusters?

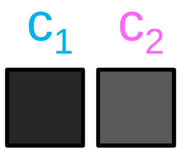
– if we know the number of clusters
  → k-means algorithm

1. initialize *k* prototype colors $c_1, c_2, \ldots, c_k$ randomly (e.g. by randomly picking pixels from image)
2. assign each pixel to the prototype color that is most similar
3. recalculate prototype colors by averaging over colors of pixel which have been assigned in step 2
4. repeat steps 2 and 3 until convergence (i.e. the assignments in step 2 do not change any more)



example: *k=2*

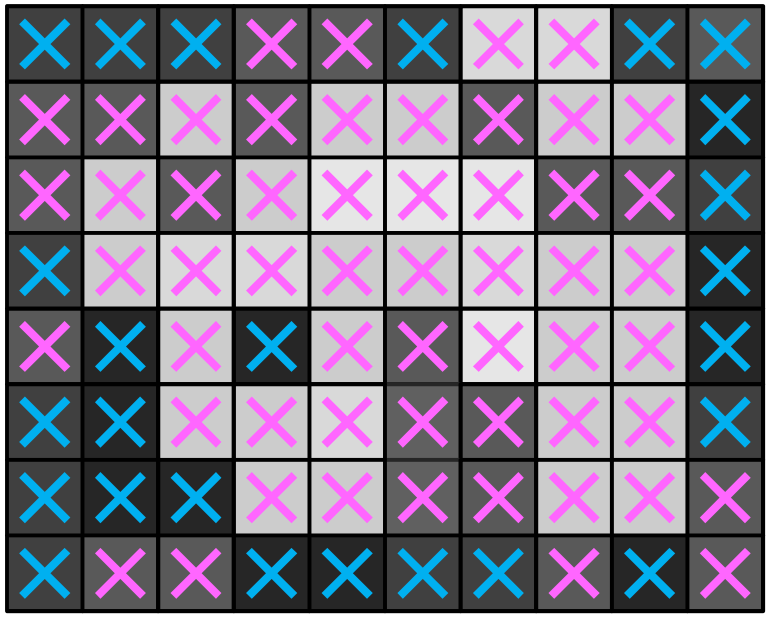
step 1: randomly pick colors from two pixels



k-means

- how can we find color clusters?
- if we know the number of clusters
  → k-means algorithm

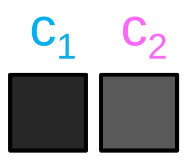
1. initialize *k* prototype colors $c_1, c_2, \ldots, c_k$ randomly (e.g. by randomly picking pixels from image)
2. assign each pixel to the prototype color that is most similar
3. recalculate prototype colors by averaging over colors of pixel which have been assigned in step 2
4. repeat steps 2 and 3 until convergence (i.e. the assignments in step 2 do not change any more)



example: *k=2*

step 1: randomly pick colors from two pixels

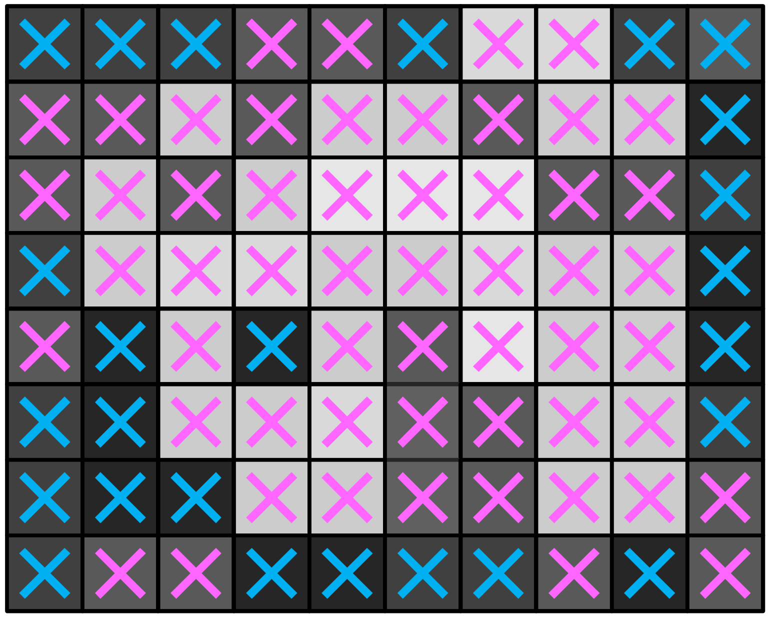
step 2: assign pixels to most similar cluster



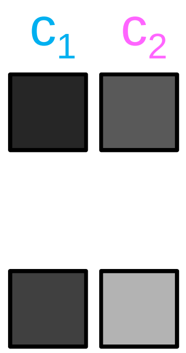
k-means

- how can we find color clusters?
- if we know the number of clusters
  → k-means algorithm

1. initialize *k* prototype colors $c_1, c_2, \ldots, c_k$ randomly (e.g. by randomly picking pixels from image)
2. assign each pixel to the prototype color that is most similar
3. recalculate prototype colors by averaging over colors of pixel which have been assigned in step 2
4. repeat steps 2 and 3 until convergence (i.e. the assignments in step 2 do not change any more)



example: *k=2*



step 1: randomly pick colors from two pixels

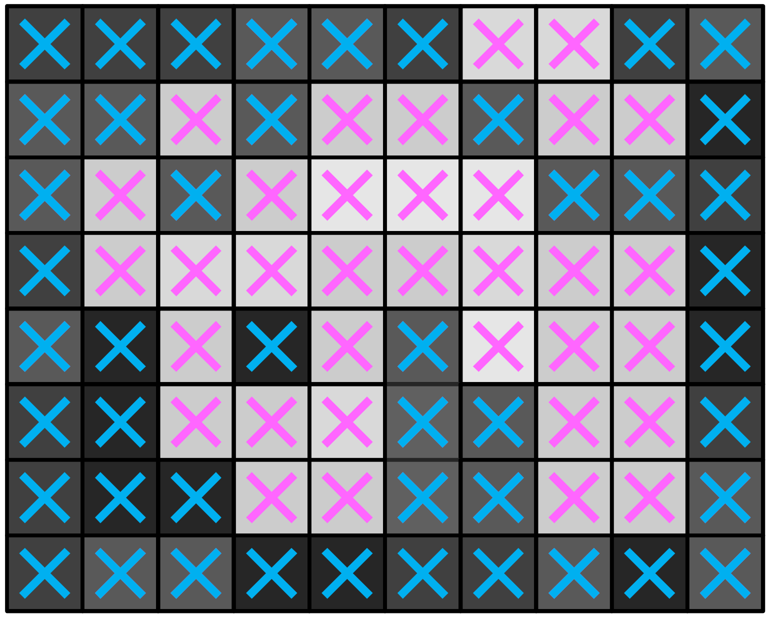
step 2: assign pixels to most similar cluster

step 3: recalculate prototype colors

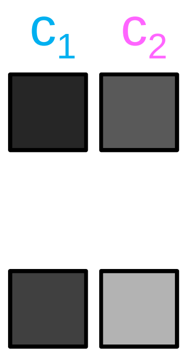
k-means

- how can we find color clusters?
- if we know the number of clusters
  → k-means algorithm

1. initialize *k* prototype colors $c_1, c_2, \ldots, c_k$ randomly (e.g. by randomly picking pixels from image)
2. assign each pixel to the prototype color that is most similar
3. recalculate prototype colors by averaging over colors of pixel which have been assigned in step 2
4. repeat steps 2 and 3 until convergence (i.e. the assignments in step 2 do not change any more)



example: *k=2*



step 1: randomly pick colors from two pixels

step 2: assign pixels to most similar cluster

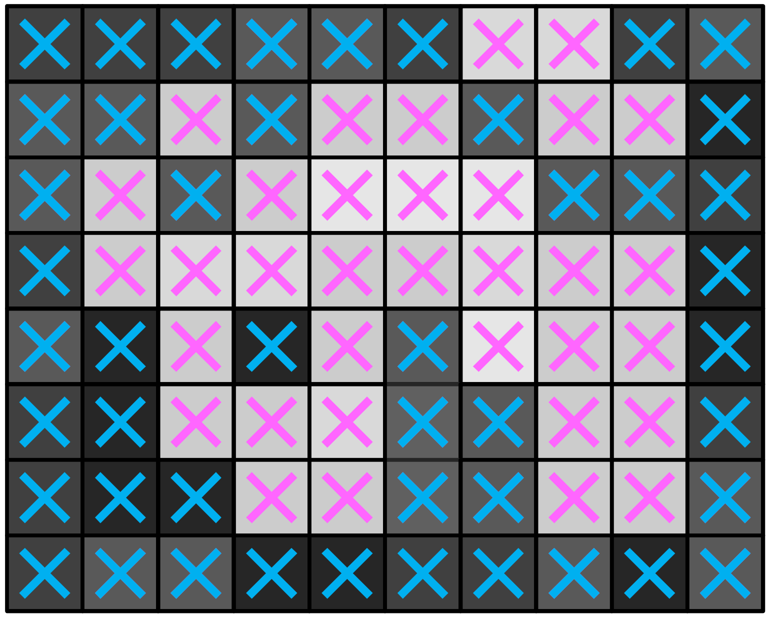
step 3: recalculate prototype colors

step 2: reassign pixels

k-means

- how can we find color clusters?
- if we know the number of clusters
  $\rightarrow$ k-means algorithm

1. initialize *k* prototype colors $c_1, c_2, \ldots, c_k$ randomly (e.g. by randomly picking pixels from image)
2. assign each pixel to the prototype color that is most similar
3. recalculate prototype colors by averaging over colors of pixel which have been assigned in step 2
4. repeat steps 2 and 3 until convergence (i.e. the assignments in step 2 do not change any more)



example: *k=2*

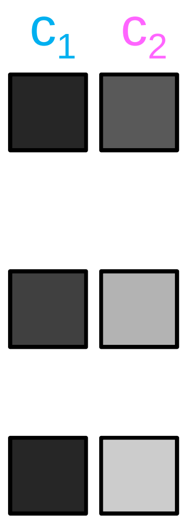
step 1: randomly pick colors from two pixels

step 2: assign pixels to most similar cluster

step 3: recalculate prototype colors

step 2: reassign pixels

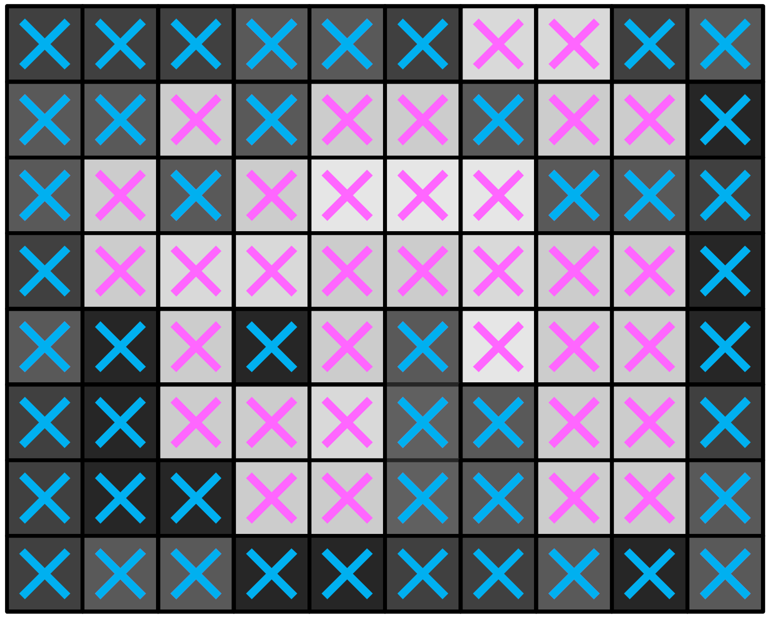
step 3: recalculate prototype colors



k-means

- how can we find color clusters?
- if we know the number of clusters
  → k-means algorithm

1. initialize *k* prototype colors $c_1, c_2, \ldots, c_k$ randomly (e.g. by randomly picking pixels from image)
2. assign each pixel to the prototype color that is most similar
3. recalculate prototype colors by averaging over colors of pixel which have been assigned in step 2
4. repeat steps 2 and 3 until convergence (i.e. the assignments in step 2 do not change any more)



example: *k=2*

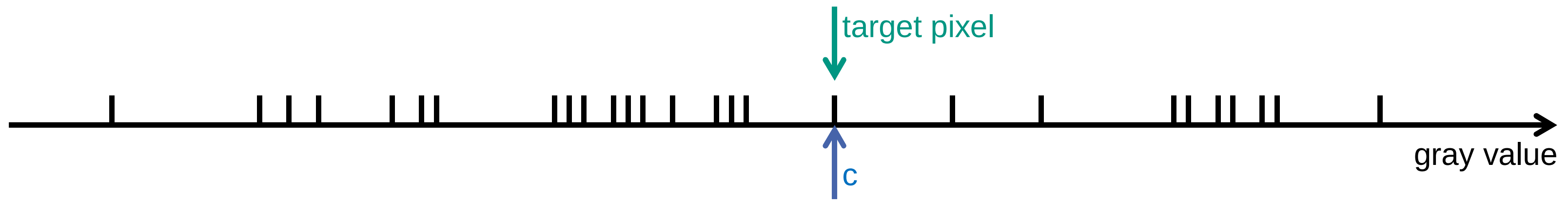
	c_1	c_2
step 1: randomly pick colors from two pixels	■	■
step 2: assign pixels to most similar cluster		
step 3: recalculate prototype colors	■	■
step 2: reassign pixels		
step 3: recalculate prototype colors	■	□
step 2: reassign pixels → convergence		

Slide 28

Mean-shift

– example

arranged all pixel colors (gray values) along one axis

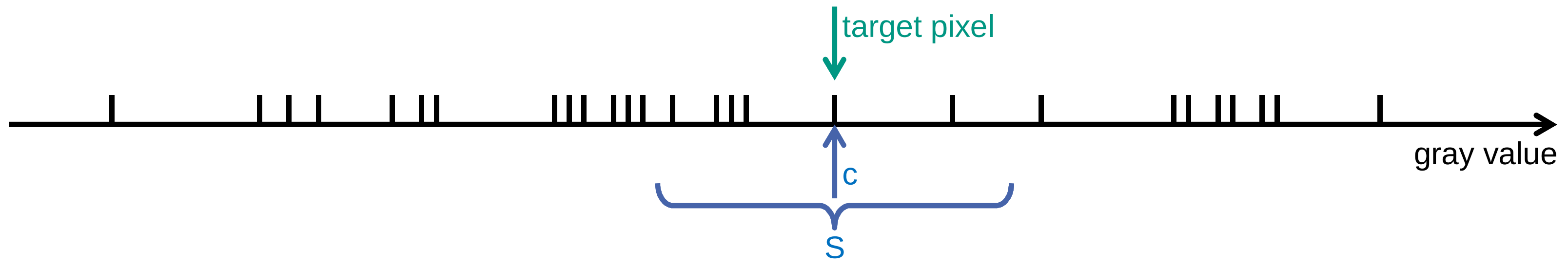


step 1: pick color of target pixel *c*

Mean-shift

– example

arranged all pixel colors (gray values) along one axis

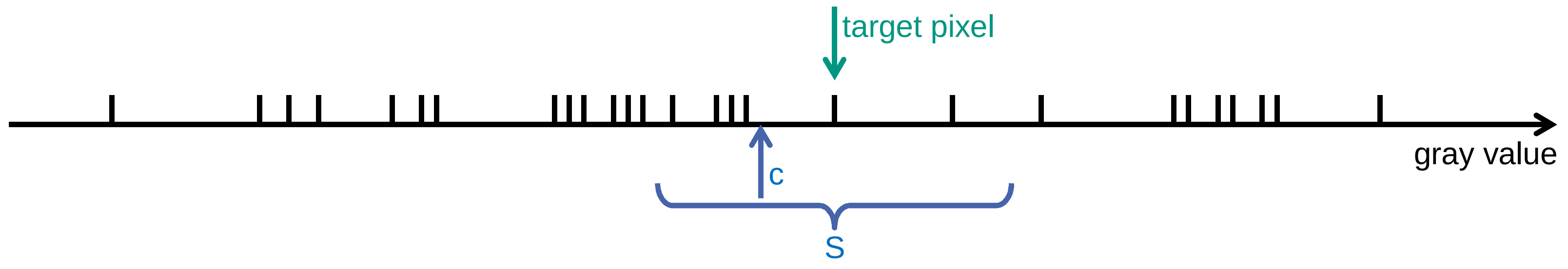


step 1: pick color of target pixel *c*
step 2: find the set of similar pixels *S*

Mean-shift

– example

arranged all pixel colors (gray values) along one axis

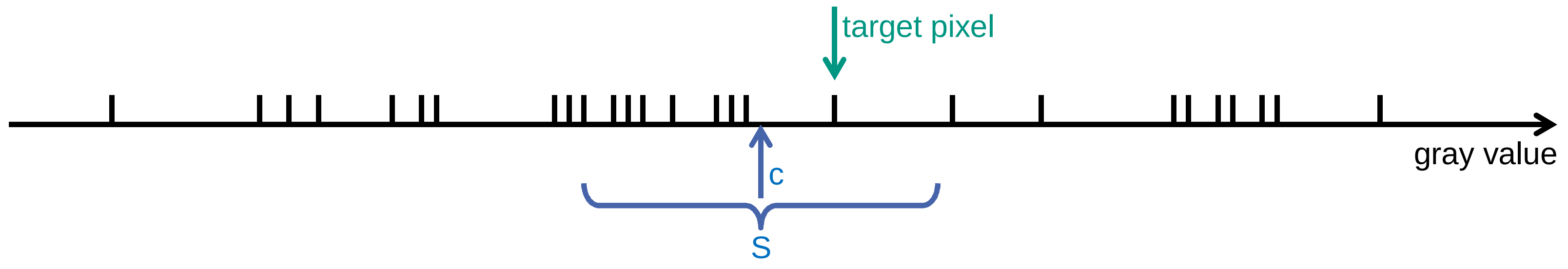


step 1: pick color of target pixel *c*
step 2: find the set of similar pixels *S*
step 3: calculate average color of *S* and assign it to *c*

Mean-shift

– example

arranged all pixel colors (gray values) along one axis

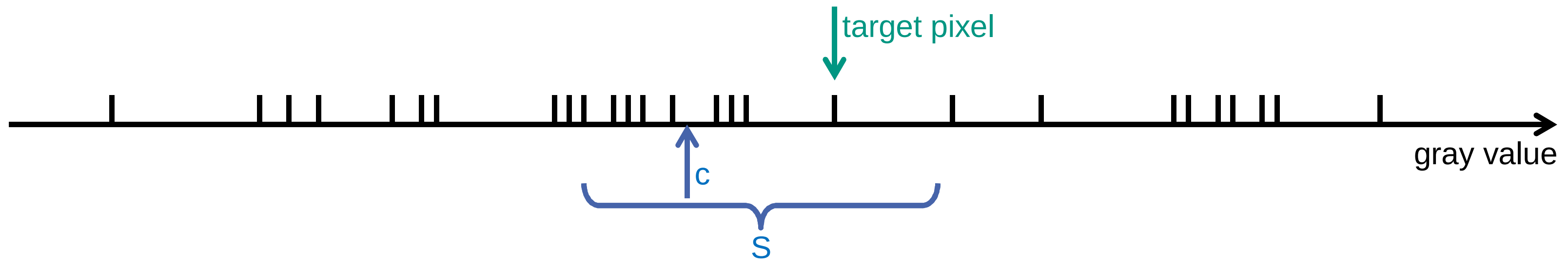


step 1: pick color of target pixel *c*
step 2: find the set of similar pixels *S*
step 3: calculate average color of *S* and assign it to *c*
step 2: recalculate *S*

Mean-shift

– example

arranged all pixel colors (gray values) along one axis

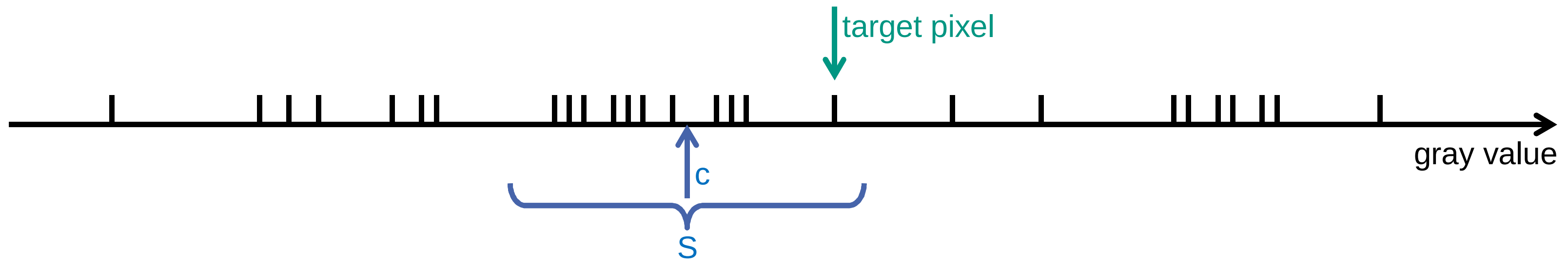


step 1: pick color of target pixel *c*
step 2: find the set of similar pixels *S*
step 3: calculate average color of *S* and assign it to *c*
step 2: recalculate *S*
step 3: recalculate average color of *S* and assign it to *c*

Mean-shift

– example

arranged all pixel colors (gray values) along one axis

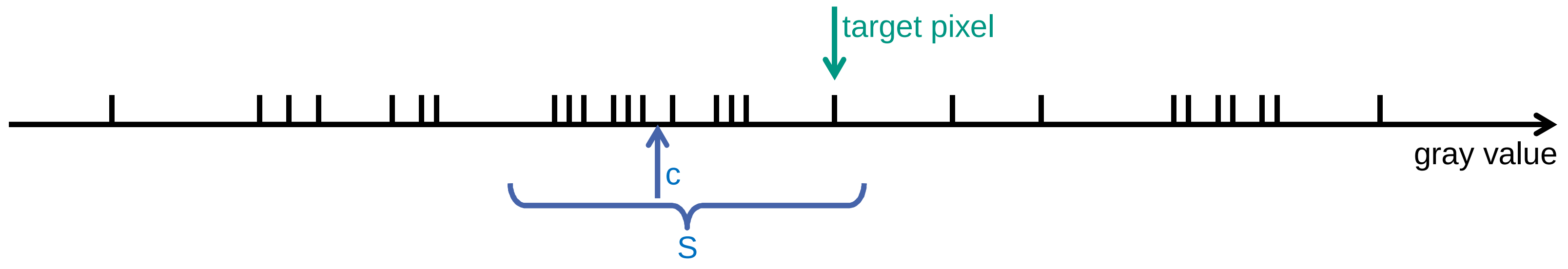


step 1: pick color of target pixel *c*
step 2: find the set of similar pixels *S*
step 3: calculate average color of *S* and assign it to *c*
step 2: recalculate *S*
step 3: recalculate average color of *S* and assign it to *c*
step 2: recalculate *S*

Mean-shift

– example

arranged all pixel colors (gray values) along one axis

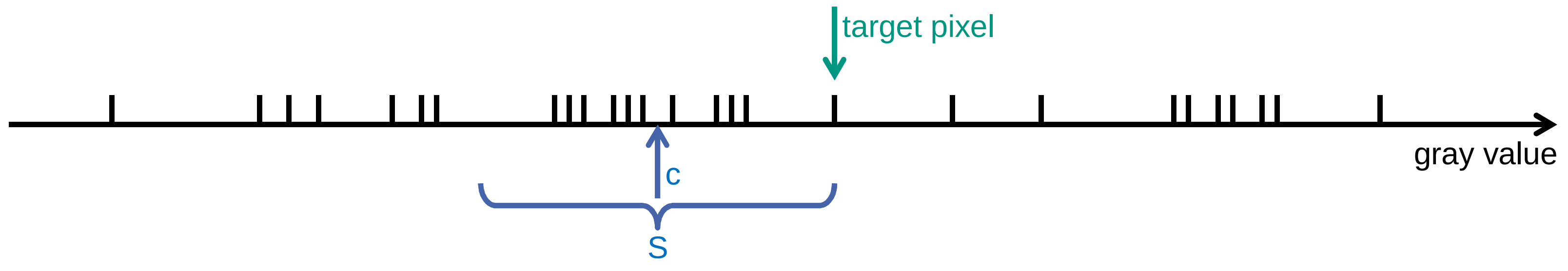


step 1: pick color of target pixel *c*
step 2: find the set of similar pixels *S*
step 3: calculate average color of *S* and assign it to *c*
step 2: recalculate *S*
step 3: recalculate average color of *S* and assign it to *c*
step 2: recalculate *S*
step 3: recalculate average color of *S* and assign it to *c*

Mean-shift

– example

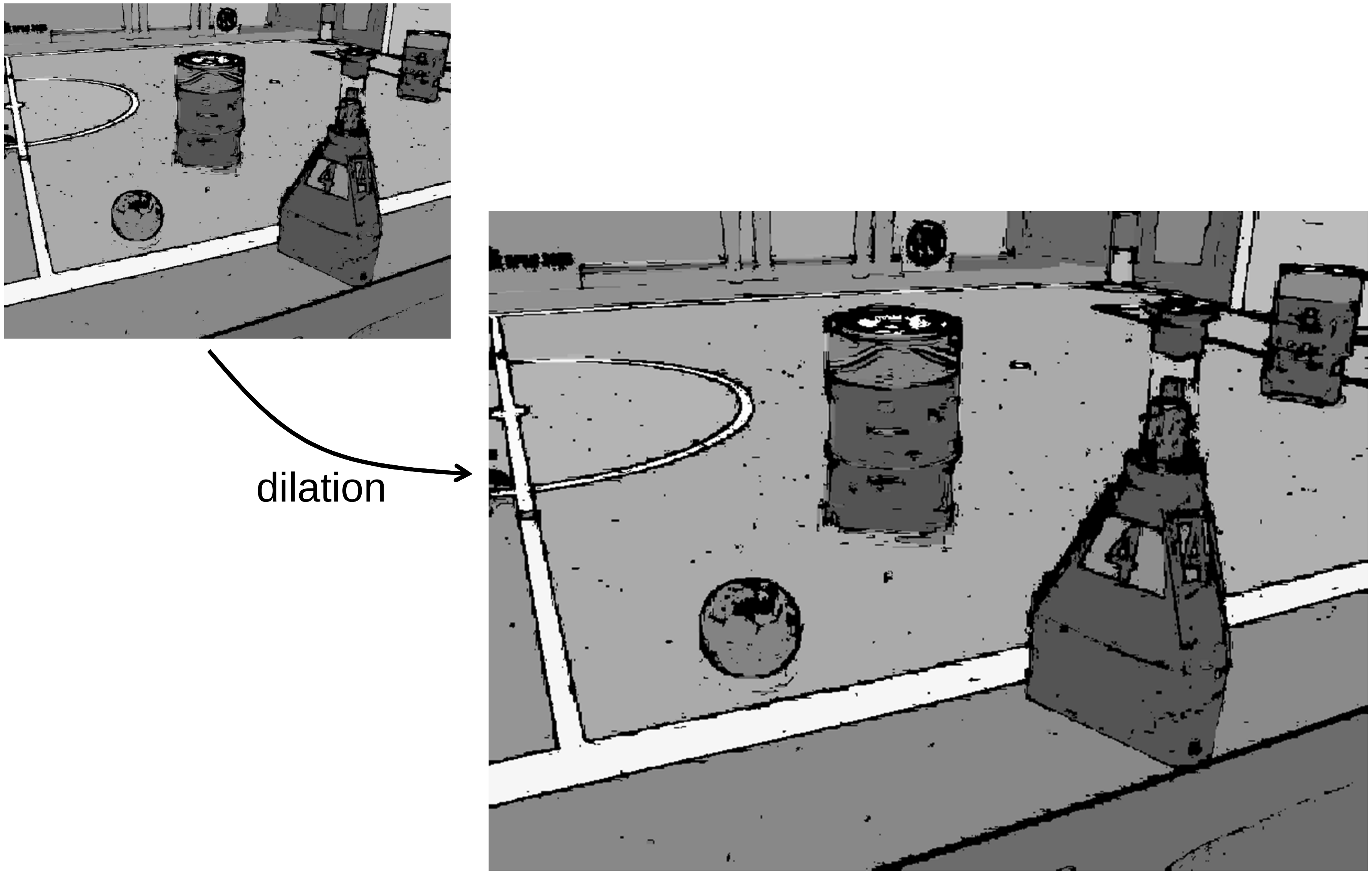
arranged all pixel colors (gray values) along one axis



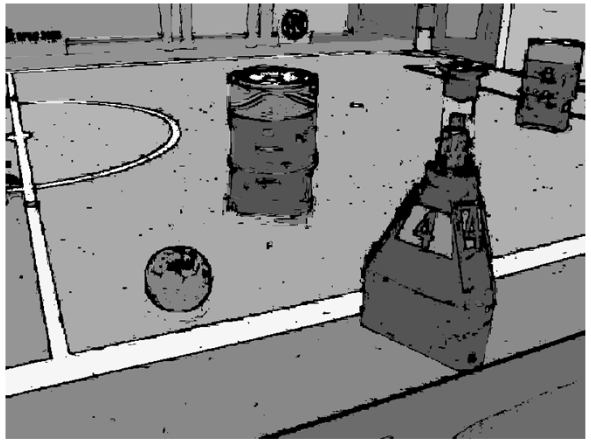
step 1: pick color of target pixel *c*
step 2: find the set of similar pixels *S*
step 3: calculate average color of *S* and assign it to *c*
step 2: recalculate *S*
step 3: recalculate average color of *S* and assign it to *c*
step 2: recalculate *S*
step 3: recalculate average color of *S* and assign it to *c*
step 2: recalculate *S* → convergence

Slide 32

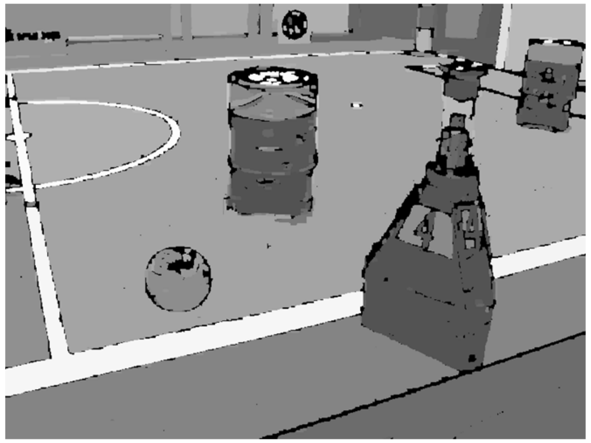
Morphological Operations cont.



Morphological Operations cont.

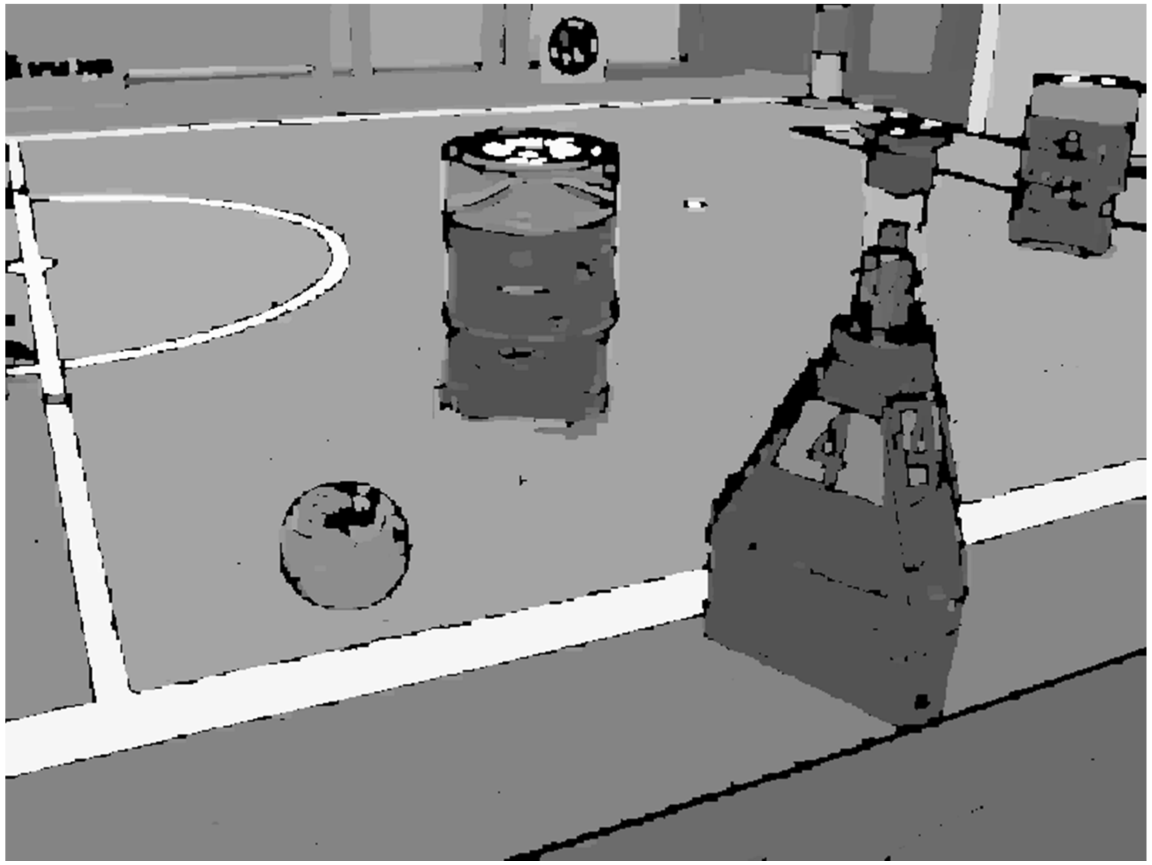


↓dilation



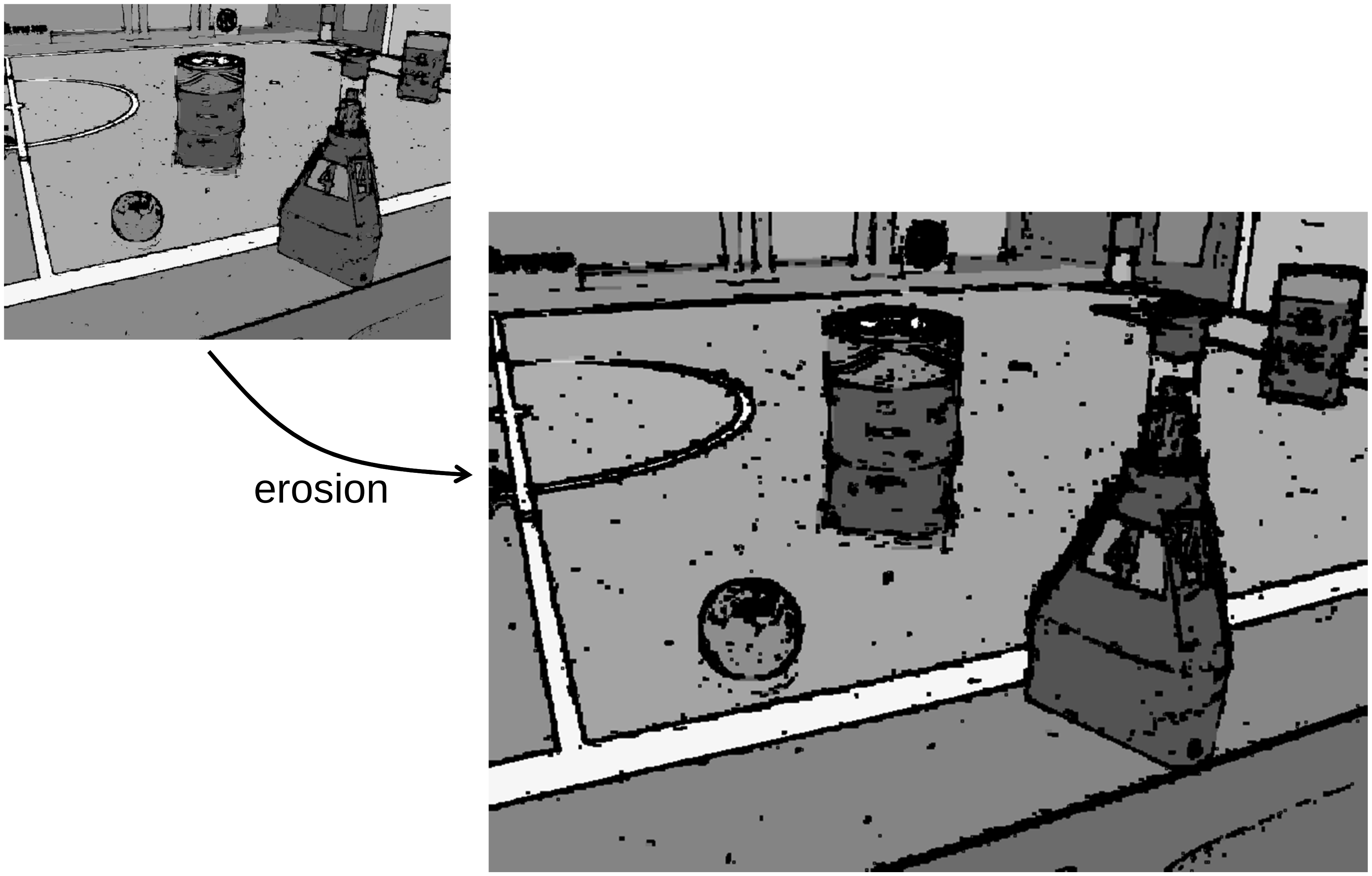
dilation

dilation fills holes and gaps

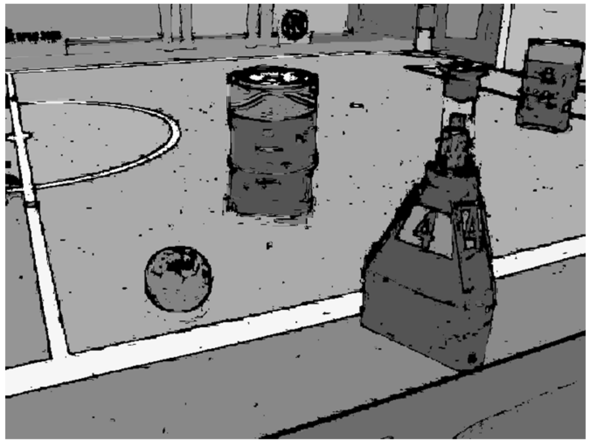


Slide 33

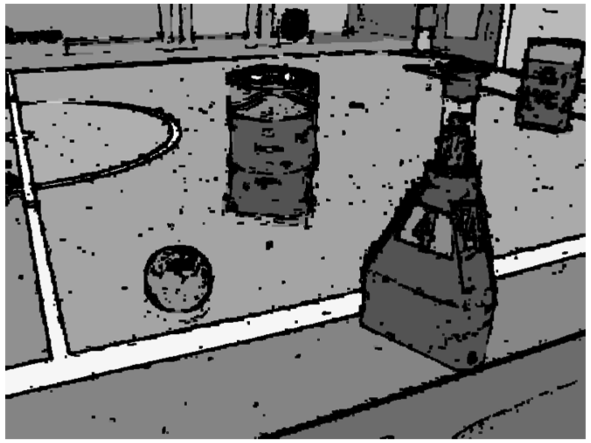
Morphological Operations cont.



Morphological Operations cont.

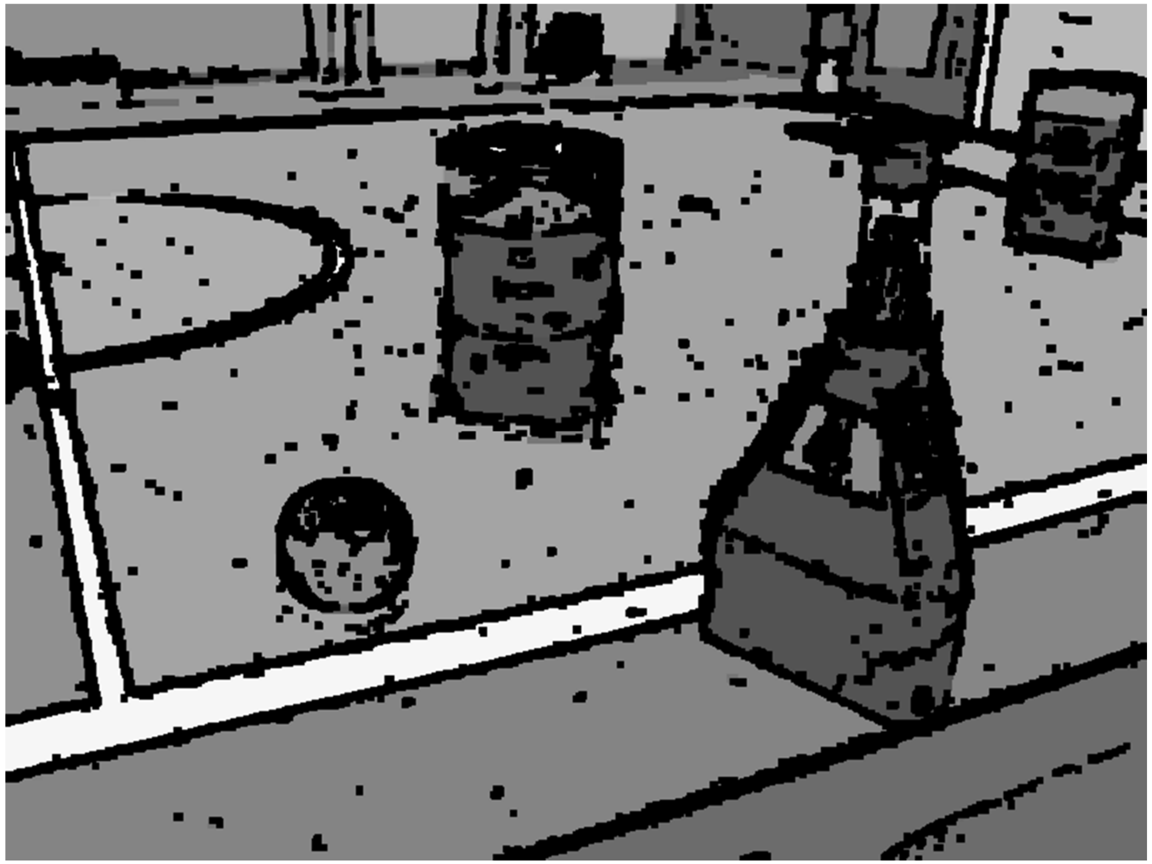


↓erosion



erosion

erosion eliminates thin structures



Slide 63

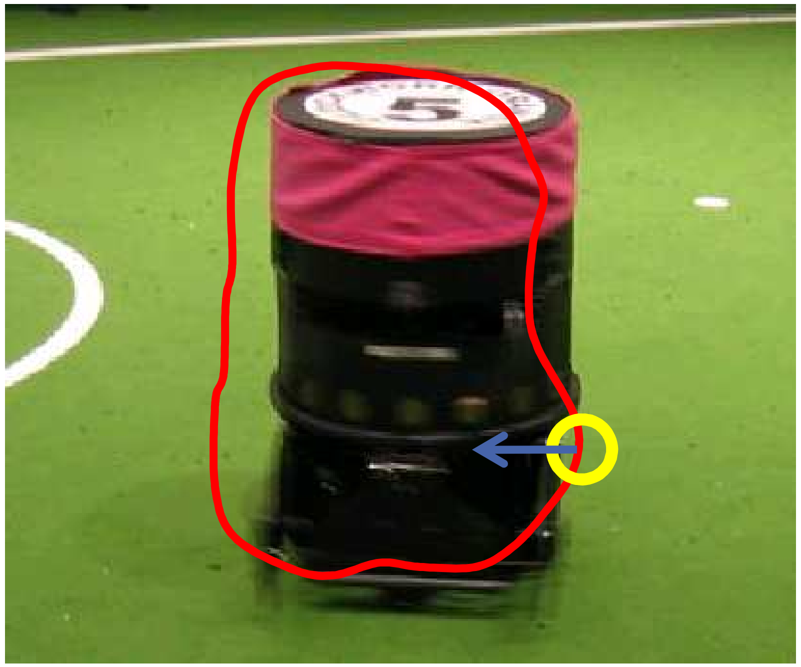
Image Segmentation with Level Sets cont.

– check for pixels on boundary with grey (color) value I

- pixel more similar to area outside

  $$(g - \bar{g}_{foreground})^2 > (g - \bar{g}_{background})^2$$

  $\rightarrow$ shrink contour



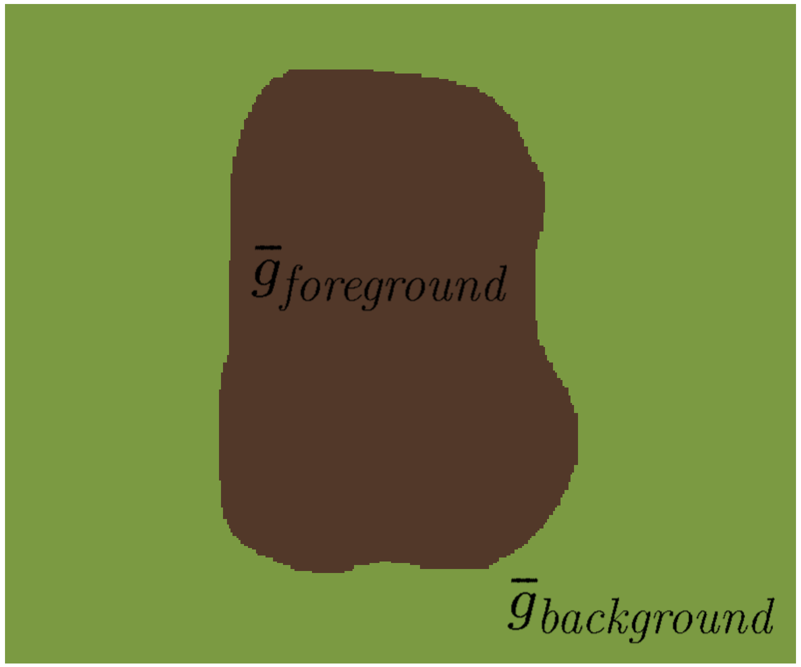


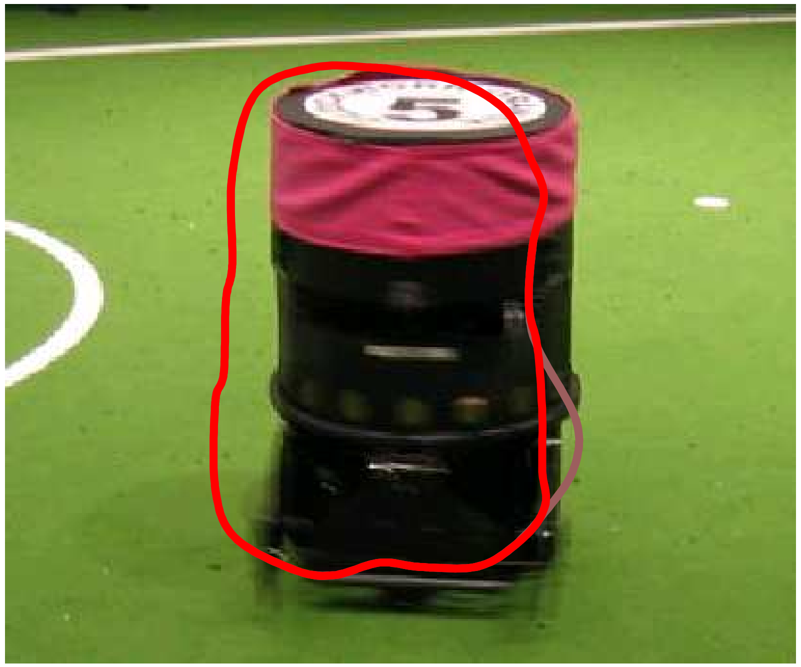
Image Segmentation with Level Sets cont.

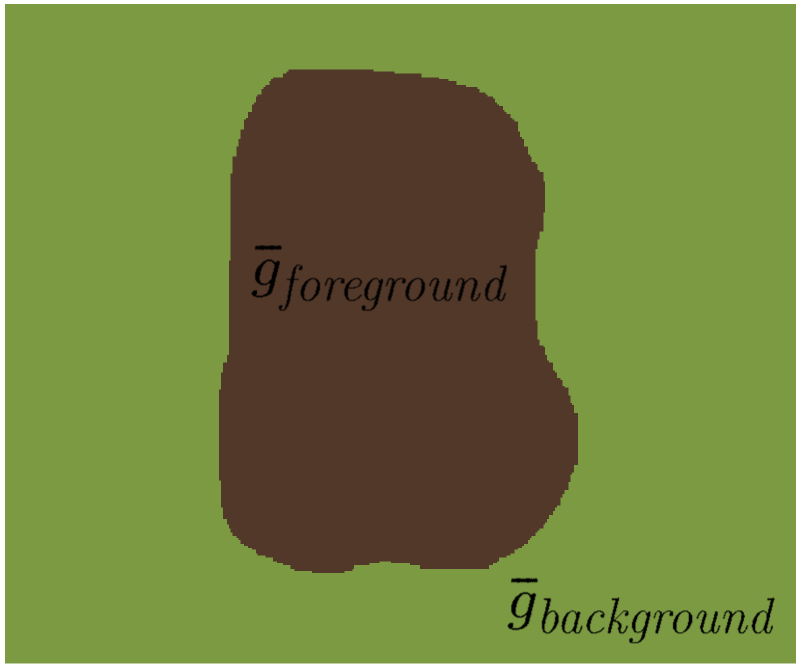
– check for pixels on boundary with grey (color) value I

- pixel more similar to area outside

$$(g - \bar{g}_{foreground})^2 > (g - \bar{g}_{background})^2$$

$\rightarrow$ shrink contour





Slide 64

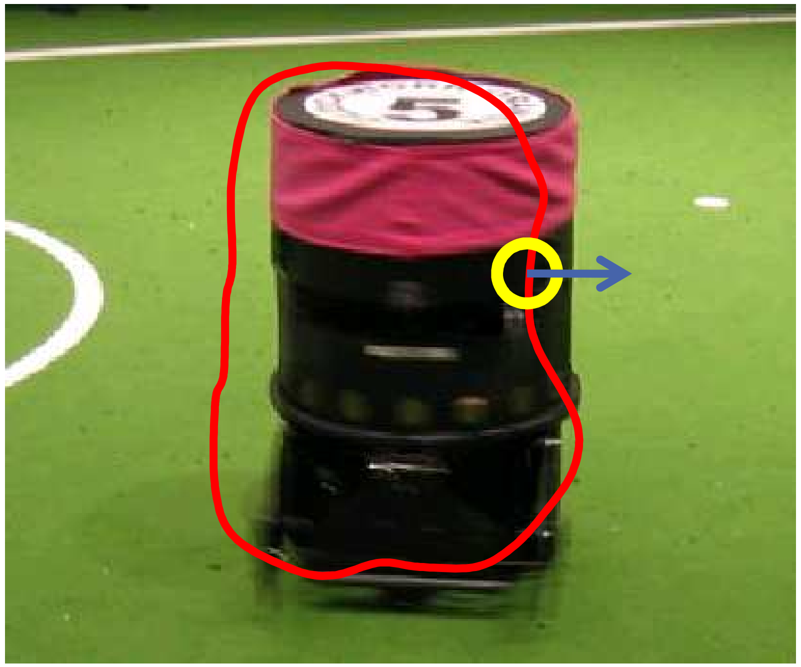
Image Segmentation with Level Sets cont.

– check for pixels on boundary with grey (color) value I

- pixel more similar to area inside

$$(g - \bar{g}_{foreground})^2 < (g - \bar{g}_{background})^2$$

$\rightarrow$ expand contour



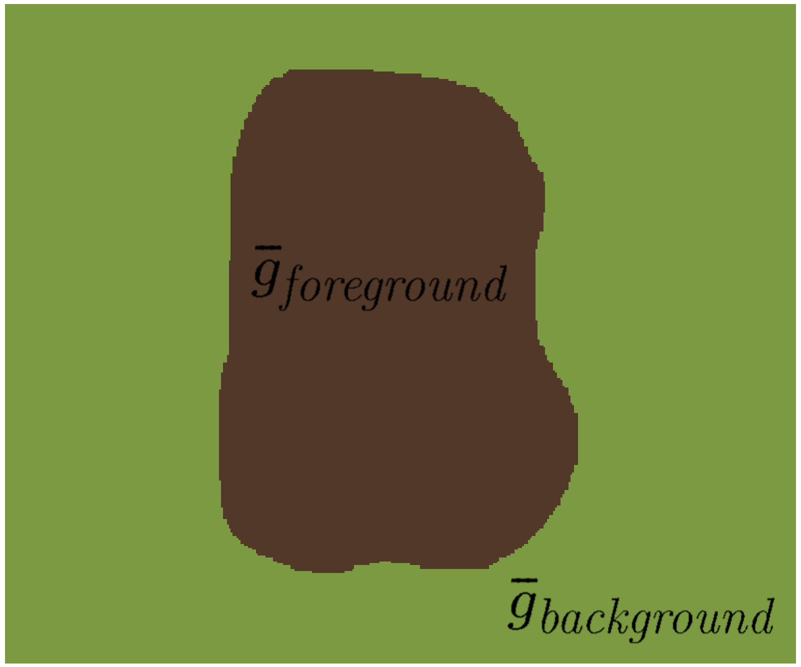


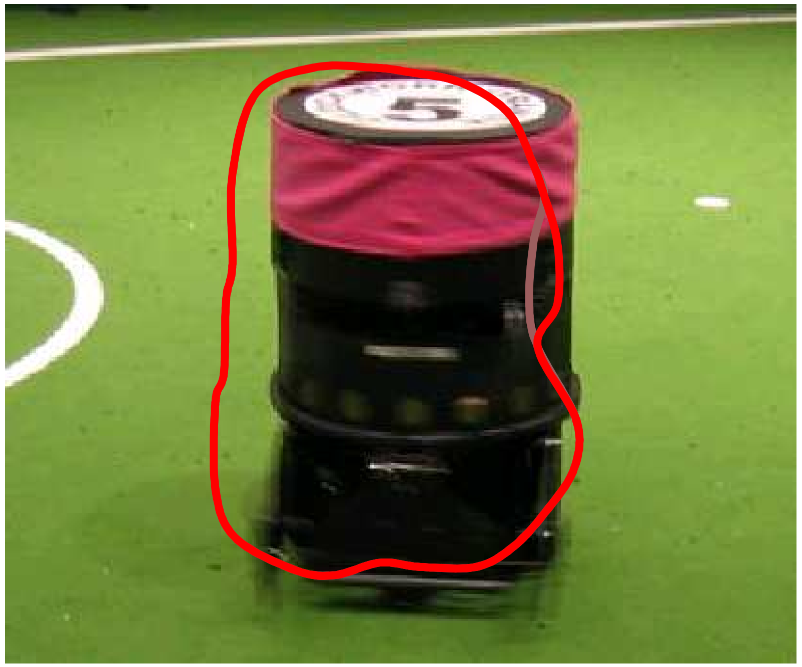
Image Segmentation with Level Sets cont.

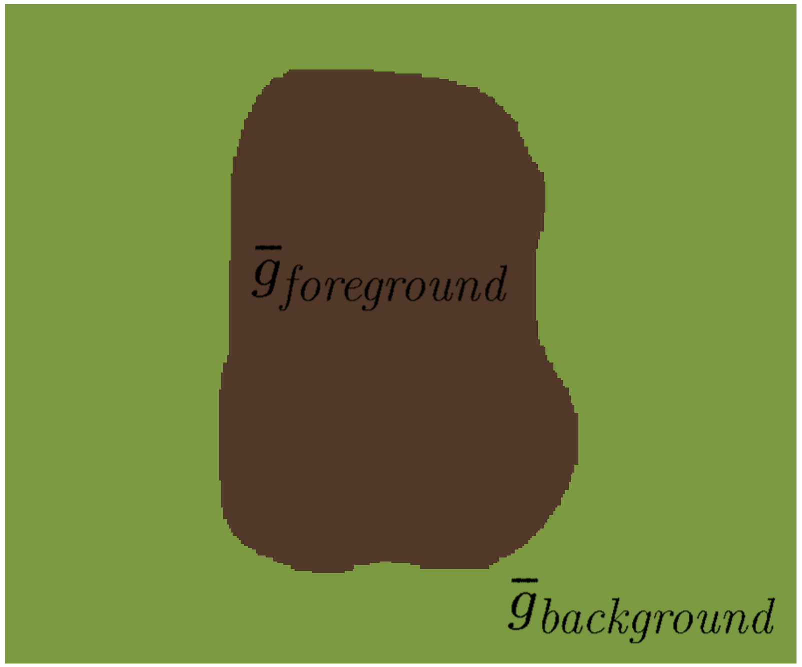
– check for pixels on boundary with grey (color) value I

- pixel more similar to area inside

  $(g - \bar{g}_{foreground})^2 < (g - \bar{g}_{background})^2$

  $\rightarrow$ expand contour




$\bar{g}_{foreground}$
$\bar{g}_{background}$