CS 331
Computer Vision

09 / 16 / 2024

Instructor: Michael Eckmann

Today's Topics

- Questions / Comments?
- Last comments on the cross-correlation worksheet from last time
- Binary Image Morphology operations and uses
- Start Connected Components

Uses of Binary Morphology

- Let me motivate at least one use of binary morphology with this image of just the green hue selected pixels

Uses of Binary Morphology

- Let's look at that a portion of that image zoomed in --- notice "holes" in the leaves

Uses of Binary Morphology

- Binary morphology can close those holes in an automated way

Morphology

- Morphology in general works on binary images (those that truly have pixels of only 1 of 2 possible colors (e.g. black or white))
- We can use a certain morphology operation on the image on the previous slide if we consider the black pixels as black (off/background) and the green pixels as white (on/foreground).
- Besides filling in holes, other morphology operations are also used to smooth jagged edges of regions, remove small regions entirely, and other purposes
- Think of the white pixels as foreground (objects) and the black pixels as background.
- 0=black=off=background (all used interchangeably)
- 1=white=on=foreground (all used interchangeably)
- If the black and white image is actually stored as grayscale (e.g. 0 – 255 values) then white would be 255.

structuring element

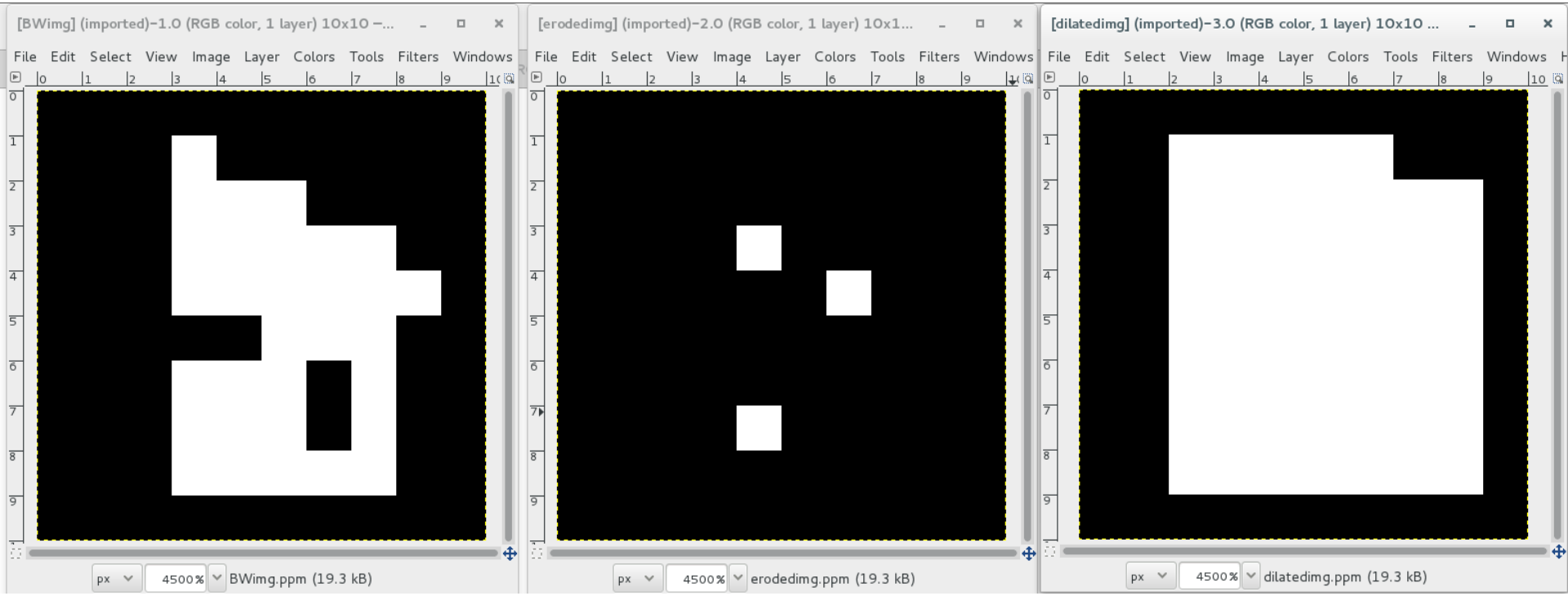
- a structuring element is typically a small binary image representing some shape. A structuring element has an origin.
  - e.g. a 3x3 square of all 1-pixels, with center as origin
  - a 5x5 binary image with a diamond shape of 1-pixels, others 0's

- a structuring element is applied to a binary image by hovering the origin pixel over each pixel in the image one at a time
  - only the 1-pixels in the structuring element matter (the 0-pixels of the structuring element are not compared to anything)
  - if the structuring element **fits,** this means that *all* the 1-pixels in the structuring element correspond to 1-pixels in the image
  - if the structuring element **hits,** this means that *at least one* 1-pixel in the structuring element corresponds to a 1-pixel in the image

binary morphology

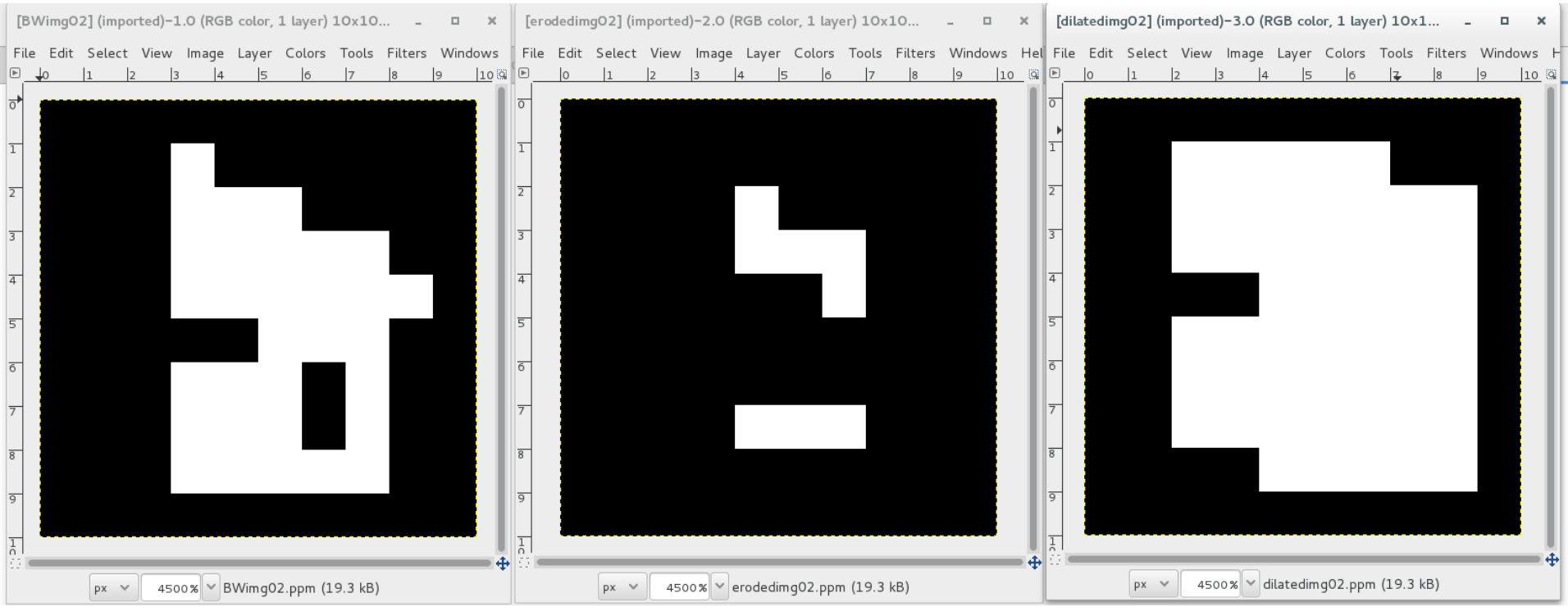
- binary image morphological operations
  - image is **f**, structuring element is **s**, resulting image is **g**
  - **dilation** – increases the size of regions
    - f dilated by s results in g where g contains a 1 where s hits f, 0 otherwise
  - **erosion** – decreases the size of regions
    - f eroded by s results in g where g contains a 1 where s fits f, 0 otherwise

examples



- Left image – original 10x10 w/ white starting in 2nd row, 4th column,
- Middle image – eroded w/ 3x3 Box,
- Right image – dilated w/ 3x3 Box.

examples



- Left image – original 10x10 w/ white starting in 2nd row, 4th column,
- Middle image – eroded w/ 3x3 L shape,
- Right image – dilated w/ 3x3 L shape.

binary morphology

- binary image morphological operations
  - Notice that dilation filled in holes, but the region size grew
  - And erosion got rid of jutting out pixels, but also the region shrunk

binary morphology

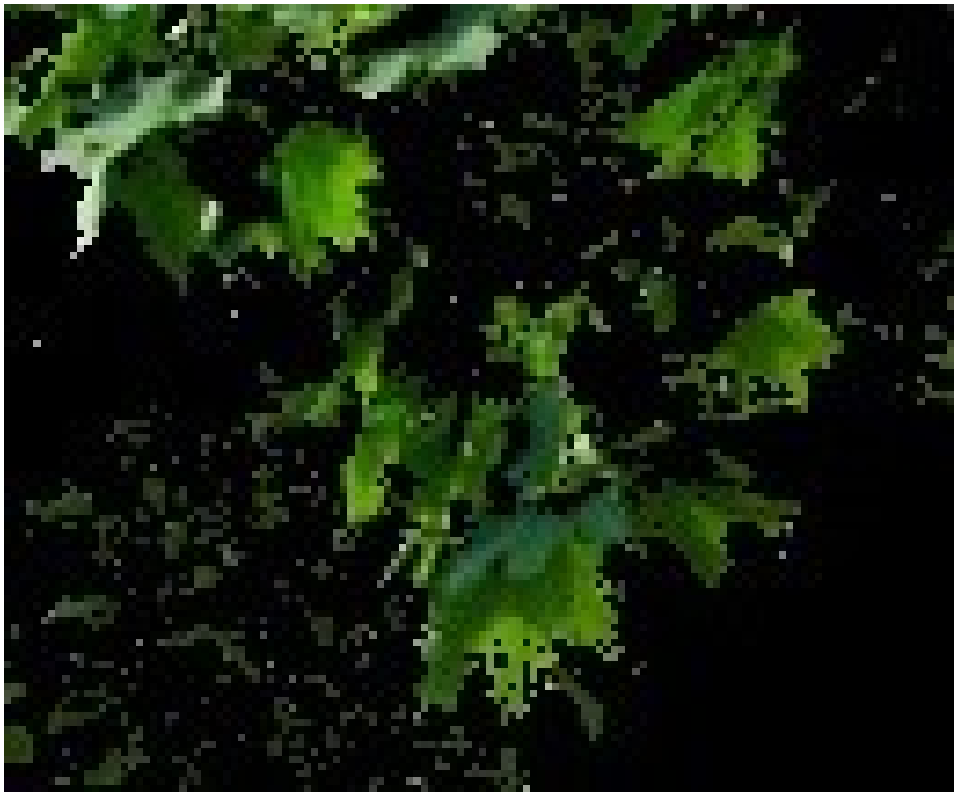
- binary image morphological operations
  - image is **f**, structuring element is **s**, resulting image is **g**
  - **closing** – closes up holes within regions
    - f closed by s results in g, which is the result of f dilated by s and then eroded by s
    - Dilate then Erode
  - **opening** – get rid of jutting out portions of regions
    - f opened by s results in g, which is the result of f eroded by s and then dilated by s
    - Erode then Dilate
- Both opening and closing are idempotent
  - this means after one closing of f by s, further closings by s do not change the result
  - same for opening

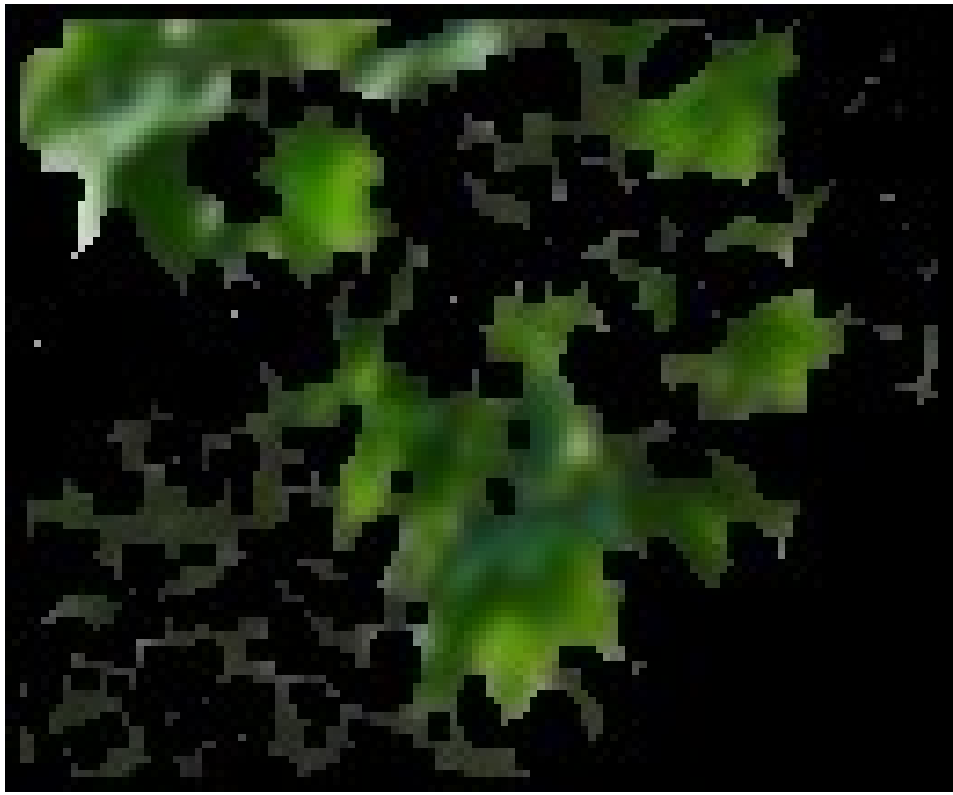
binary morphology

- binary image morphological operations
  - The result of closing would be to keep the region about the same size while filling in gaps
  - The result of opening would be to keep the region about the same size while removing jutting out pixels
  - For that green image let's close it with a 3x3 structuring element of all 1's --- I tweaked the idea of binary morphology for this --- if an output pixel was to be written as "on" or "white", I made it the average of the 8 "on" neighbors. That is I averaged the nearby greens and itself.

binary morphology

- Result (on the right) of closing the left image with a 3x3 all 1's structuring element



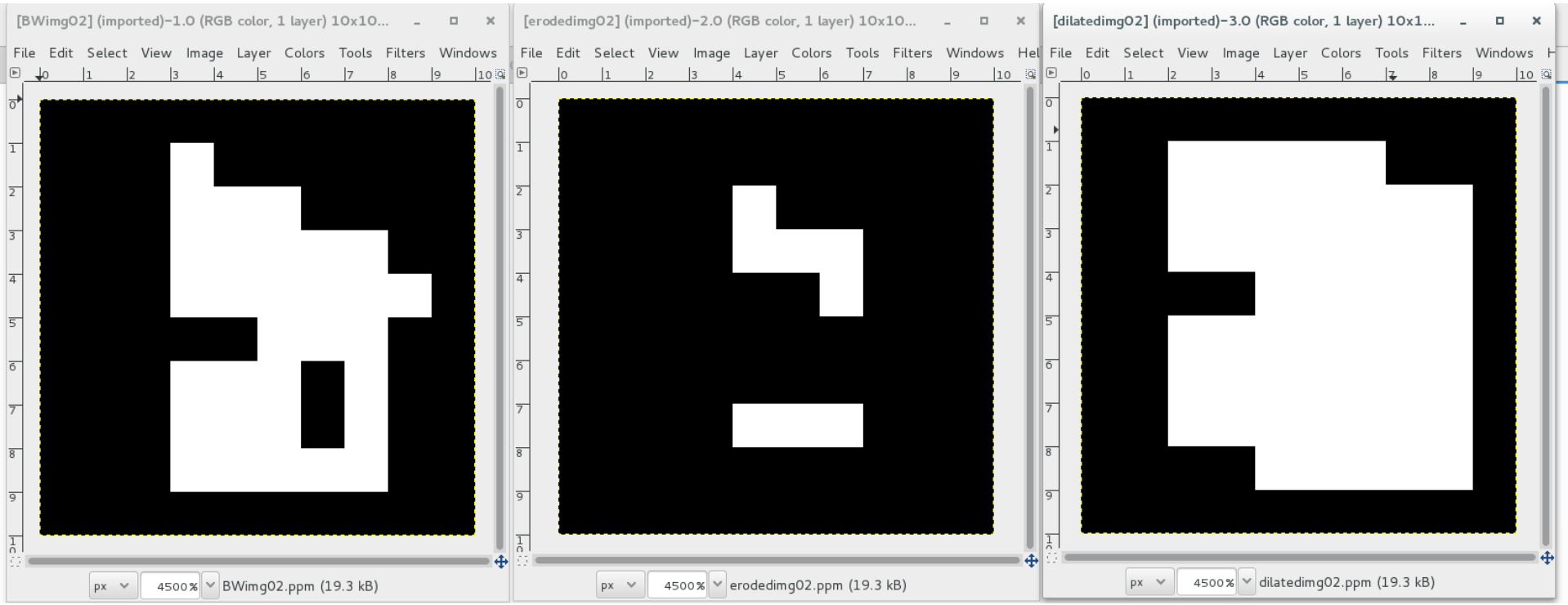


binary morphology

- Other simple operations that can be done to binary images
  - f and g are binary images of the same dimensions
  - AND: f AND g results in h where h contains a 1 iff f and g both contain a 1
  - OR: f OR g results in h where h contains a 1 if either f and/or g contain a 1
  - NOT: NOT f results in h where all 0's in f result in 1's in h and all 1's in f result in 0's in h
  - MINUS: f MINUS g results in h where
    - 1 minus 0 = 1
    - all others 0

Connected components

- Once we have cleaned up our image using morphology, we then might like to figure out which pixels belong to which regions.
- A pixel belonging to a region means it is connected to other pixels in the same region.
- Left and right images below contain only 1 region, middle image contains 2.



Connected component problem

- Problem:
  - Give a label to each connected component of an image (consider black/white images only for now where black is background, white is foreground). A connected component is a set of connected foreground pixels.

Connected components

- We can consider either neighbors as 4-adjacent or 8-adjacent
  - 4-adjacent means a pixel is connected if it is adjacent to another pixel in the region either up, down, left or right
  - 8-adjacent means a pixel is connected if it is adjacent to another pixel in the region either up, down, left, right or diagonally up/left, up/right, down/left, down/right