Middlebury

CSCI 146: Intensive Introduction to Computing

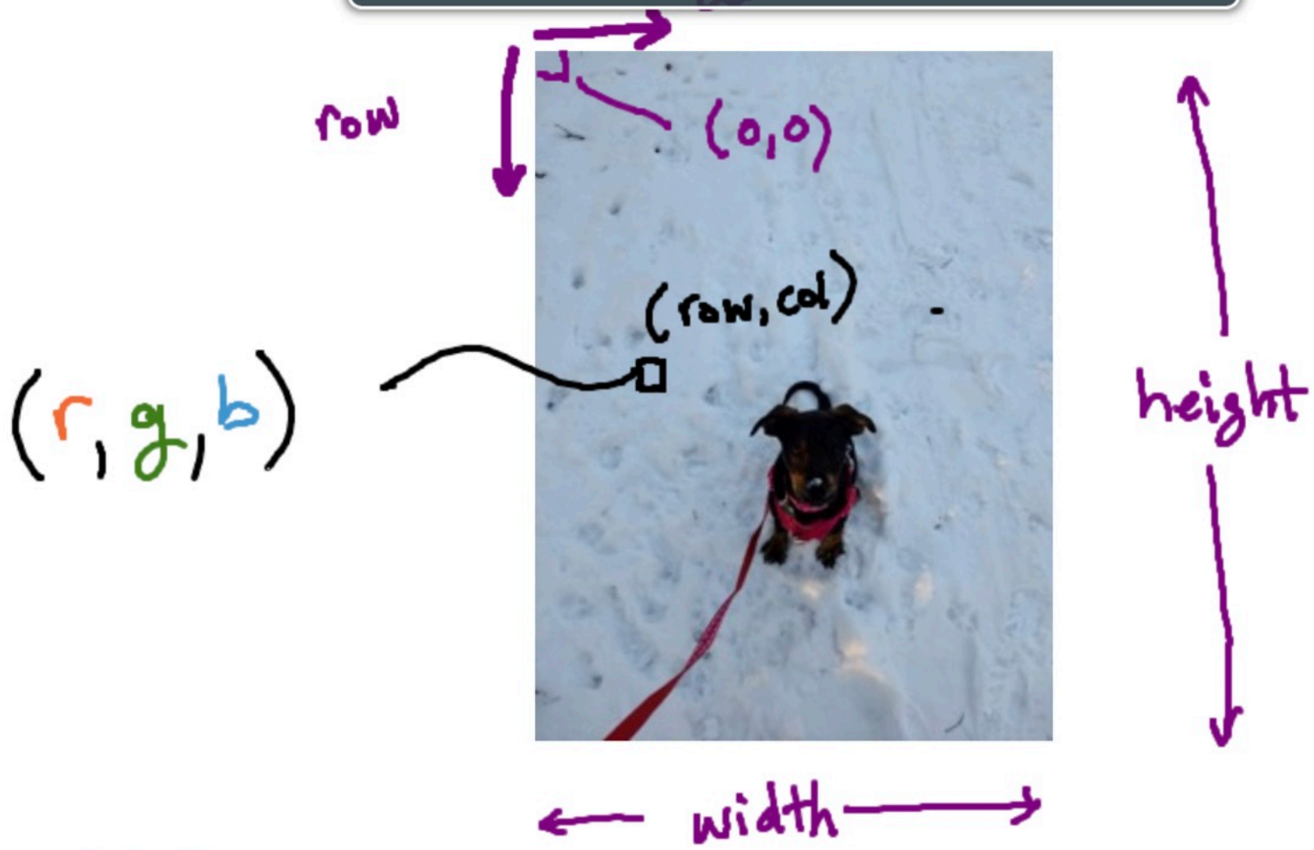
Fall 2025

Lecture 21: Image Processing

Goals for today

- Implement nested loops to perform "2-D" computations
- Utilize an existing helper class for building a program
- Implement some common image processing techniques

Representing images using a 2D array of pixels.



We'll use `middimage`:

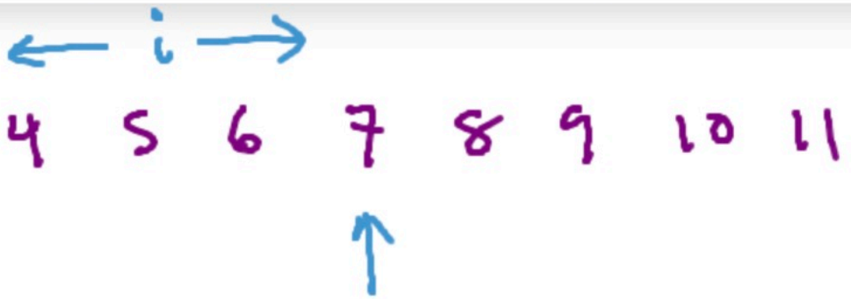
```
>>> from middimage import *
>>> img = Image("puppy-front.jpg")
>>> pixel = img.get_pixel(0, 0) # top-left Pixel
```

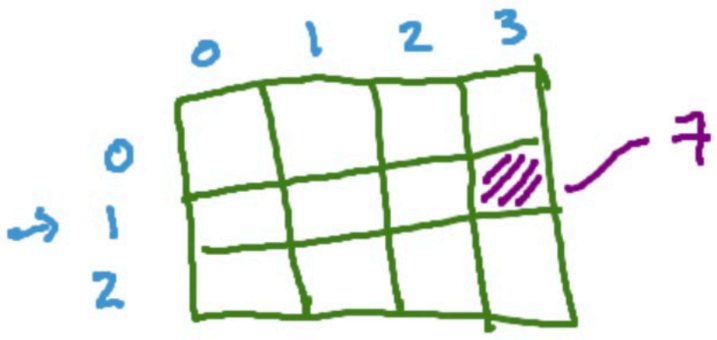
Indexing 2D structures in a linear (1D) way.

```
ROWS = 3
COLS = 4

for row in range(ROWS):
    for col in range(COLS):
        print("Row:", row, "Col:", col, "Linear index:", row * COLS + col)
```

Going the other way: what are the 2D (row, column) indices from a unique 1D index?

```
ROWS = 3
COLS = 4

for i in range(ROWS * COLS):
    row = i // COLS
    col = i % COLS
    print("Row:", row, "Col:", col, "Linear index:", i)
```

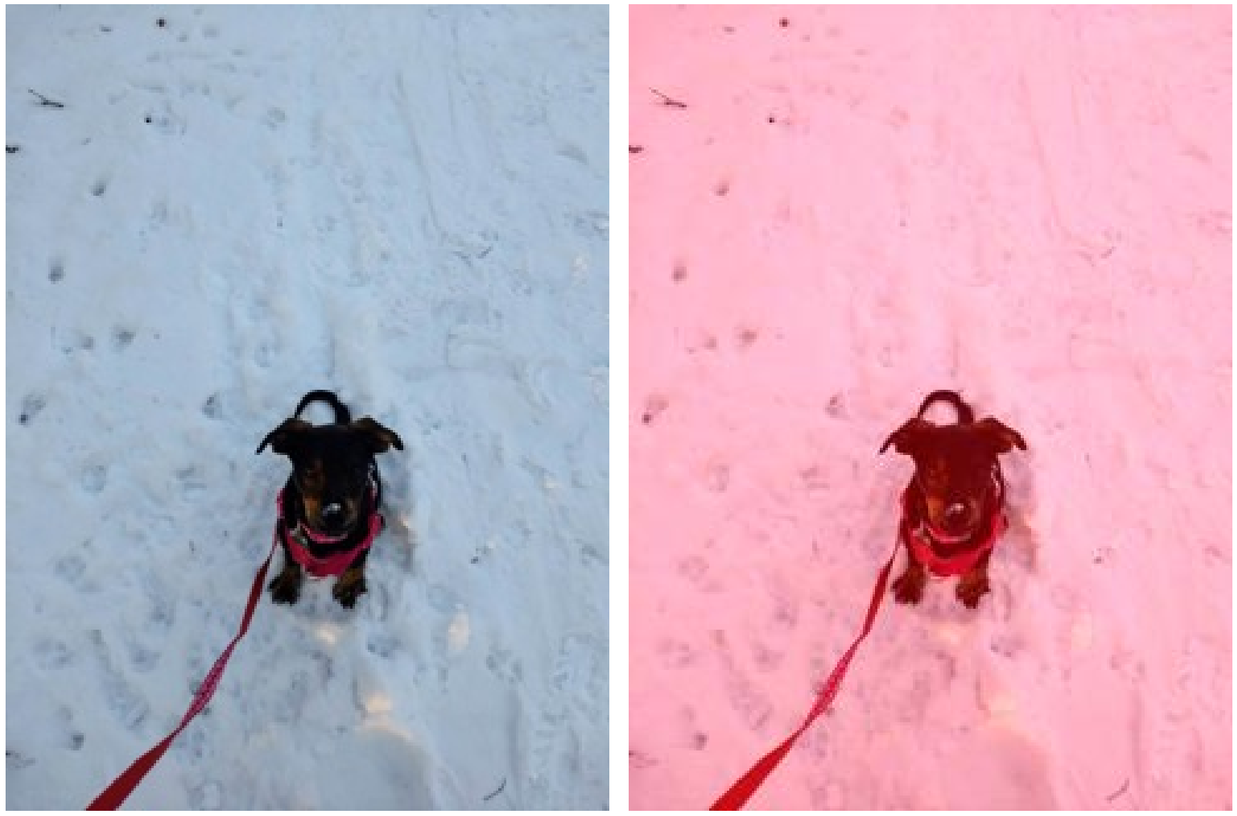




7 // 4 = 1

7 % 4 = 3

Red shift: add to red component of input image pixel colors.



Use the `get_width()` and `get_height()` methods of `Image` object to define range of loops.

Inverse: each output pixel color is (255, 255, 255) minus input pixel color.

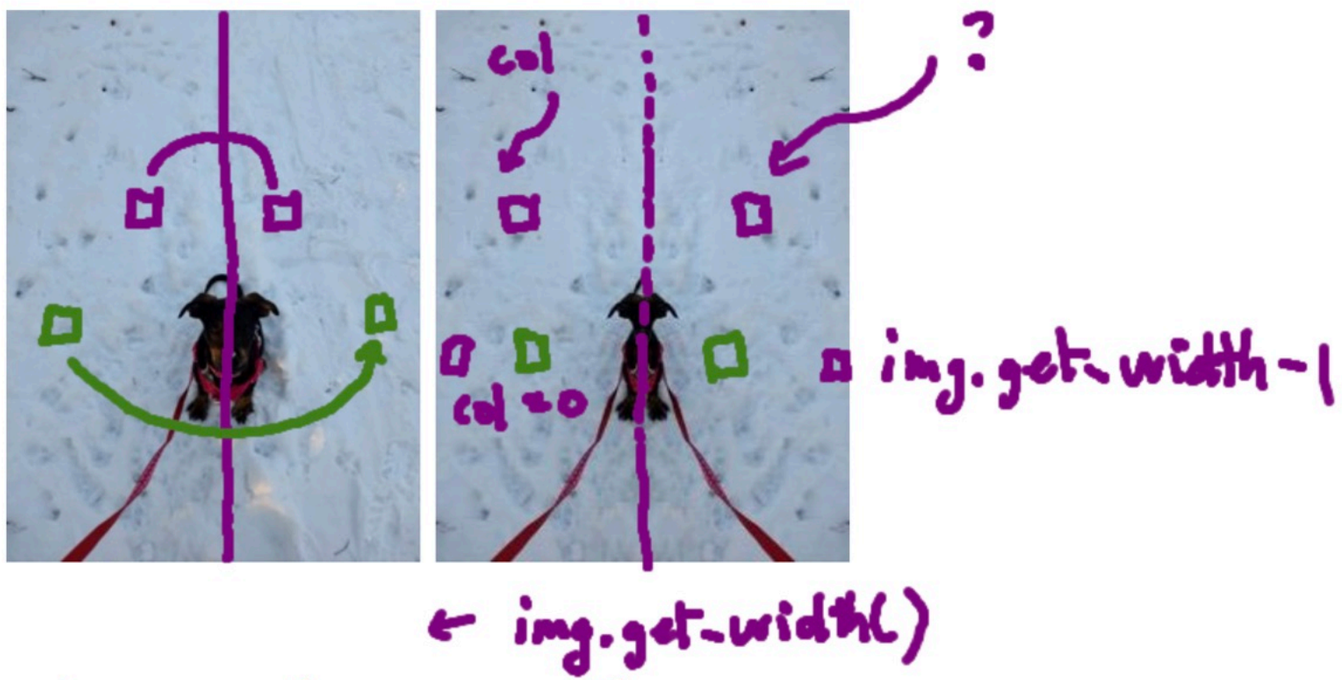
Your turn: complete the `sepia` function.

$$R^* = 0.393R + 0.769G + 0.189B$$

$$B^* = 0.349R + 0.686G + 0.168B$$

$$G^* = 0.272R + 0.534G + 0.131B$$

Mirroring an image across a vertical line in the center.



A pixel on the left half of `img` with `(row, col)` will be copied to a pixel with a column index of (`col` starts at `0`):

A. `img.get_width() + col`

B. `img.get_width() - col`

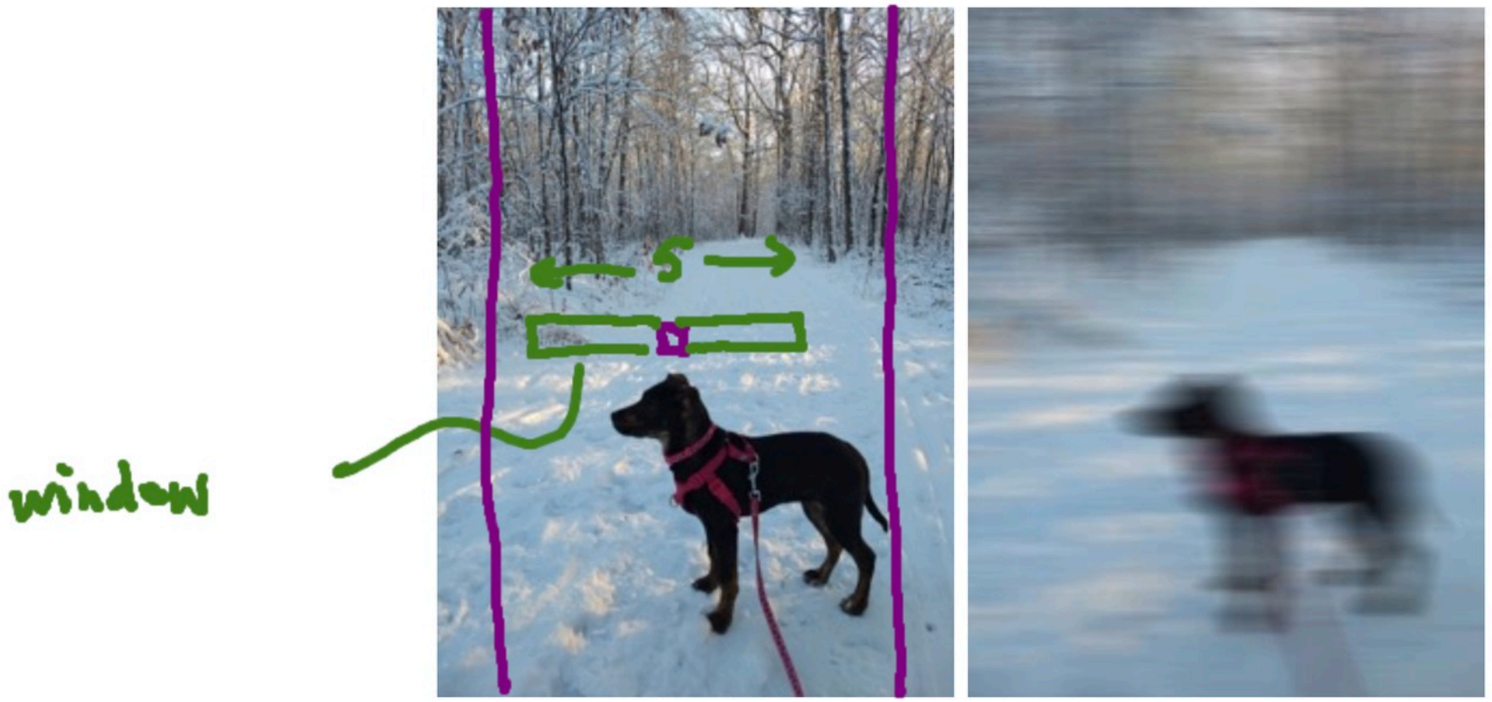
C. `img.get_width() - col - 1`

D. `img.get_width() // 2 + col - 1`

Loops within loops within a loop. Blurring an image using a `window` of pixels to average.

We'll average `window` pixels centered on this pixel (in the same row). For example, if `window` is 5, we'll use this pixel along with the 2 pixels to the left and 2 pixels to the right.

What kinds of things do we need to consider (and be careful of)? boundaries



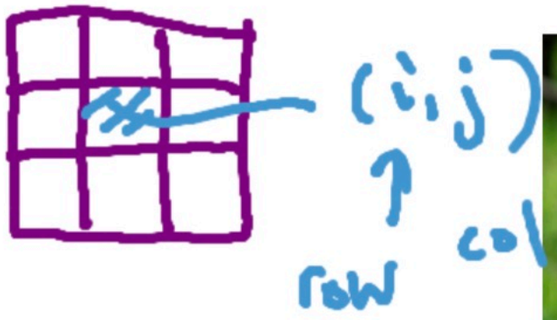
Exercise: detecting edges in two steps: (1) convert to grayscale, (2) apply filter

$$a_{i,j} = (p_{i-1,j+1} + 2p_{i,j+1} + p_{i+1,j+1}) - (p_{i-1,j-1} + 2p_{i,j-1} + p_{i+1,j-1})$$

$$b_{i,j} = (p_{i-1,j-1} + 2p_{i-1,j} + p_{i-1,j+1}) - (p_{i+1,j-1} + 2p_{i+1,j} + p_{i+1,j+1})$$

$$d_{i,j} = \sqrt{a_{i,j}^2 + b_{i,j}^2}$$

output pixel grayscal



Have a look at some of the operators that are overloaded in the `Pixel` class!

Reminders

- Self-scheduled **Exam retakes** on Friday 12/5: question a, b, c, d, e, f, g, h.
- Self-scheduled **Quiz retakes** on Monday 12/8: quizzes 1 - 9
- Final review on Monday 12/8
- **Test Project** is available (initial due date 12/8, final due date: 12/15).
  - No collaboration.
  - No Google searching or trying to find any parts of the solution online.
  - Please treat this as a "programming test". More details on the website.
- Programming Assignment 8 final due date this Thursday.
- Programming Assignment 9 initial due date this Friday.