Lecture 7:
Upsampling
A whirlwind tour of numpy
Announcements

• Project 1 out Very Soon(TM)
  
  • i.e., by the end of the weekend
Goals

• Know how to upsample images naively

• Know how to upsample images using reconstruction filters.

• Know the basics of how to use the numpy library
Reconstruction

\[
L_0 \quad \quad L_1 \quad \quad L_2 \quad \quad L_3 \quad \quad L_4 \quad \quad L_5
\]

\[
\text{levels} = \text{reverse}(\text{levels})
\]
\[
\text{img} = \text{levels}[0]
\]
\[
\text{for } i \text{ in } 1..\text{len(levels)}:
\]
\[
\quad \text{img} = \text{upscale}_2x(\text{img})
\]
\[
\quad \text{img} += L_i
\]
Upsampling

• But how do we make images bigger?

• Again: a naive way and a principled way.

```python
def upscale_2x(img):
    # Implementation of upscale_2x function
    return 

levels = reverse(levels)
img = levels[0]
for i in 1..len(levels):
    img = upscale_2x(img)
    img += Li
```
Upsampling

- This image is too small for my screen. How do I make it 10x bigger?
Upsampling

- This image is too small for my screen. How do I make it 10x bigger?

- Simple approach: repeat each row and column 10 times
Upsampling: Interpolation

- Another way to look at this: we need to double the sampling rate.
Upsampling: Interpolation

- Another way to look at this: we need to double the *sampling rate*.
- But we don't actually know the continuous function:
Upsampling: Nearest Neighbor
Upsampling: Linear
Upsampling: Linear

A filtering perspective

![Graph showing upsampling process]

\( x \)

\( \frac{1}{2} \)

\( \frac{3}{2} \)
Upsampling: Nearest Neighbor

A filtering perspective
Upsampling Filters in 2D

1D: \( h \)

2D: "tent filter"
Upsampling by 4X

1. Make 4Hx4W image of zeros.
2. Fill in every 4th pixel.
3. Filter*! (and multiply by 16)
numpy

• Tutorials:
  • https://numpy.org/devdocs/user/quickstart.html
  • https://cs231n.github.io/python-numpy-tutorial/#numpy

• Demo!

• Exercises
Demo!

• Feel free to follow along

    ssh -p 922 username@labs.cs.wwu.edu
    wget https://facultyweb.cs.wwu.edu/~wehrwes/courses/csci497p_20s/lectures/L07_np/van.png
    ipython3
    import numpy as np

• Demo and image files at:

    • https://facultyweb.cs.wwu.edu/~wehrwes/courses/csci497p_20s/lectures/L07_np/
Exercises!

• Also available at

1. Suppose a is a filter and b is a patch of an image:
   
   ```python
   a = np.array([[1, 2, 1],
                 [2, 4, 2],
                 [1, 2, 1]]) / 16
   b = np.zeros((3,3))
   b[:,0] = 1
   b[1,1] = 2
   ```

   a. Compute the output pixel in a convolution when the filter a overlaps the image neighborhood b. Use array operations and the sum function.

   b. Compute the same product as above, but using the dot function. Hint: you'll need to reshape the inputs to dot first!

2. Load the van.png image and save out a grayscale version computed by averaging the three color channels; be sure to do the averaging in floating-point.

3a. Load the van image do a naive 2x subsampling: drop every other row and column and save out the half-size version.

3b. Load the van image and do a naive 2x upsampling: repeat every other row and column twice.