Assignment 1

- Brighten, EdgeDetect, and Blur are provided to you as examples.
- Executable reference solution is provided.
- Colors are represented as integers, with a, r, g, b each taking 1 byte.
- You can use Java's Color class, or you can use bit-wise operation:
  ```java
  int a, r, g, b;
  a=(pixel>>24)&0xff;
  r=(pixel>>16)&0xff;
  g=(pixel>> 8)&0xff;
  b=(pixel    )&0xff;
  pixel=(a<<24)+(r<<16)+(g<<8)+b;
  ```
- Image Mosaic
1D Convolution

- Moving Averages

\[ P(i) = \sum_{k=-r}^{r} \frac{1}{2r+1} I(i+k) \]
1D Convolution

- Moving Averages
- Convolution

\[ P(i) = \sum_{k=-r}^{r} \frac{1}{2r+1} I(i+k) \]

\[ P(i) = \sum_{k=-r}^{r} c_k I(i+k) \quad \left( \sum_{k=-r}^{r} c_k = 1 \right) \]
1D Convolution

- Convolution filter/kernel
  - Average filter (box filter)
    \[ c_k = \frac{1}{2r+1}, \quad -r \leq k \leq r \]
  - Convolving a step function with a box?
- Convolution of continuous function
  \[ (c \ast f)(x) = \int c(t) f(x-t) \, dt \]
Properties of Convolution

- Commutative
  \[(c \ast f)(x) = (f \ast c)(x)\]
  Why?

- Associative

- Distributive
2D Image Convolution

- Straightforward extension from 1D

\[ P(i, j) = \sum_{m,n} c_{m,n} I(i + m, j + n), \quad \sum_{m,n} c_{m,n} = 1. \]
2D Image Convolution

- Straightforward extension from 1D

\[ P(i, j) = \sum_{m, n} c_{m,n} I(i + m, j + n), \quad \sum_{m, n} c_{m,n} = 1. \]
2D Image Convolution

- Straightforward extension from 1D

\[ P(i, j) = \sum_{m,n} c_{m,n} I(i+m, j+n), \quad \sum_{m,n} c_{m,n} = 1. \]

- Boundary issue?
  - Zero boundary extension
  - Constant boundary extension
  - Keep track of partial sum of kernel weights, then renormalize after convolution
2D Image Convolution

- 2D convolution filters/kernels
  - Box filter
    \[ c_{m,n} = \frac{1}{(2r + 1)^2} \]
  - Gaussian filter
    \[ c(x, y) = \frac{1}{2\pi \sigma^2} e^{-\frac{x^2 + y^2}{2\sigma^2}} \]
  - Filter size depends on the variance \(\sigma\)
2D Image Convolution

- Gaussian Blur

Original  Blur
2D Image Convolution

- What about Sharpen?

Original  Blur
2D Image Convolution

- Sharpen (I)
  - Extrapolate from the blurred version!
2D Image Convolution

- Sharpen (II) – Equivalent to I
  - Use sharpen kernel (filter must sum up to 1)

\[
\begin{bmatrix}
0 & -1 & 0 \\
-1 & 5 & -1 \\
0 & -1 & 0
\end{bmatrix}
\]

Original

Sharpen
2D Image Convolution

- Sharpen (II) – Equivalent to I
  - Use sharpen kernel (filter must sum up to 1)

\[
\begin{bmatrix}
0 & -1 & 0 \\
-1 & 5 & -1 \\
0 & -1 & 0 \\
\end{bmatrix}
\]

\[
\frac{1}{6}
\begin{bmatrix}
0 & -1 & 0 \\
-1 & 10 & -1 \\
0 & -1 & 0 \\
\end{bmatrix}
\]
2D Image Convolution

- Sharpen (II) – Equivalent to I
  - Controllable:
    \[
    \begin{bmatrix}
    0 & -s & 0 \\
    -s & 1 + 4s & -s \\
    0 & -s & 0
    \end{bmatrix}
    \]
2D Image Convolution

- **Edge Detector**
  - Convolve with a filter that finds differences between neighbor pixels, filter weights sum up to zero
  - Essentially a Laplacian operator: 2\textsuperscript{nd} order spatial derivative
    \[
    \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} \rightarrow \begin{bmatrix}
    0 & -1 & 0 \\
    -1 & 4 & -1 \\
    0 & -1 & 0
    \end{bmatrix}
    \]
    Region of constant intensity $\rightarrow 0$

What about region of linearly changing intensity?
2D Image Convolution

- Edge detector

- Can also use Laplacian of Gaussian (LoG)
2D Image Convolution

- **Separable Filters**
  - Advantages: fast filtering

\[ c(x, y) = c(x) \cdot c(y) \]

- Many 2D filters are separable
  - Box
  - Gaussian
  - Tent
  - ...

Antialiasing in Image Processing

- Original Signal
  - Either a continuous or high-res digital signal.
Antialiasing in Image Processing

- Direct Resampling (equivalent to reduce sampling rate)
Antialiasing in Image Processing

- Aliasing
Antialiasing in Image Processing

- **General Strategy**
  - Pre-filter image signal via convolution with blur filter to **limit the high frequencies of the input signal**.
  - Resample at lower-sampling rate afterwards.

- **Rationale**
  - Human eyes prefer low-frequency blurring over high-frequency aliasing.
Antialiasing in Image Processing

- Blur the original signal
Antialiasing in Image Processing

- Resampling the blurred signal
Antialiasing in Image Processing

- Aliasing reduced
Antialiasing in Image Processing

- Aliasing reduced