Previous Lecture:
- Image processing
  - Add frame, mirror

Today's Lecture:
- More image processing
  - Color to grayscale
  - "Noise" filtering
  - Edge finding

Announcements:
- Project 4 due Thursday after spring break
- There are no office/consulting hours during spring break; they resume on Monday after the break.

Grayness: a value in [0..255]

- 0 = black
- 255 = white

These are integer values
Type: uint8

Example: color \(\rightarrow\) black and white

Can "average" the three color values to get one gray value.

Vectorized code to create a mirror image

```matlab
A = imread('LawSchool.jpg');
[nr,nc,np] = size(A);
for c = 1:nc
    B(:,c,1) = A(:,nc+1-c,1)
    B(:,c,2) = A(:,nc+1-c,2)
    B(:,c,3) = A(:,nc+1-c,3)
end
imwrite(B,'LawSchoolMirror.jpg')
```

Even more compact vectorized code to create a mirror image...

```matlab
for c = 1:nc
    B(:,c,1) = A(:,nc+1-c,1)
    B(:,c,2) = A(:,nc+1-c,2)
    B(:,c,3) = A(:,nc+1-c,3)
end
B = A(:,nc:-1:1,:)
```

Averaging the RGB values to get a gray value

\[
\text{Gray} = 0.3R + 0.59G + 0.11B
\]
Averaging the RGB values to get a gray value

\[ M(i,j) = 0.3R(i,j) + 0.59G(i,j) + 0.11B(i,j) \]

Scalar operation

\[ R \]
\[ G \]
\[ B \]

Here are 2 ways to calculate the average. Are gray value matrices \( g \) and \( h \) the same given image data \( A \)?

\[
\begin{align*}
\text{for } r = 1:nr & \\
\text{for } c = 1:nc & \\
g(r,c) &= A(r,c,1)/3 + A(r,c,2)/3 + A(r,c,3)/3; \\
h(r,c) &= (A(r,c,1)+A(r,c,2)+A(r,c,3))/3;
\end{align*}
\]

A: yes  B: no

showToGrayscale.m

Matlab has a built-in function to convert from color to grayscale, resulting in a 2-d array:

\[ B = \text{rgb2gray}(A) \]

Clean up “noise” — median filtering

Dirt in the image!

Note how the “dirty pixels” look out of place

150 149 152 153 152 155 151 150 153 154 153 156 153 156
156 154 158 159 158 161 156 154 158 159 156 159
157 156 159 160 159 162
Assign "typical" neighborhood gray values to "dirty pixels"

Assign typical neighborhood gray values to "dirty pixels"

What to do with the dirty pixels?

What are "typical neighborhood gray values"?

Median
Mean
radius 1
radius 2

Median Filtering

- Visit each pixel
- Replace its gray value by the median of the gray values in the "neighborhood"

What are "typical neighborhood gray values"?

Median
Mean
radius 1
radius 2

Using a radius 1 "neighborhood"

Before

After

What we need…

1. A function that computes the median value in a 2-dimensional array C:
   \[ m = \text{medVal}(C) \]

2. A function that builds the filtered image by using median values of radius r neighborhoods:
   \[ B = \text{medFilter}(A,r) \]

Visit every pixel; compute its new value.

for \( i=1:m \)
  for \( j=1:n \)
    Compute new gray value for pixel \((i,j)\).
  end
end

What we need…

1. A function that computes the median value in a 2-dimensional array C:
   \[ m = \text{medVal}(C) \]

2. A function that builds the filtered image by using median values of radius r neighborhoods:
   \[ B = \text{medFilter}(A,r) \]
Computing the median

\[ x : \begin{array}{cccccc}
21 & 89 & 36 & 28 & 19 & 88 & 43 \\
\end{array} \]

\[ x = \text{sort}(x) \]

\[ x : \begin{array}{cccccc}
19 & 21 & 28 & 36 & 43 & 88 & 89 \\
\end{array} \]

\[
\begin{align*}
\text{n} &= \text{length}(x); \quad \% \text{n} = 7 \\
\text{m} &= \text{ceil}(n/2); \quad \% \text{m} = 4 \\
\text{med} &= x(\text{m}); \quad \% \text{med} = 36
\end{align*}
\]

If \( n \) is even, then use:

\[
\text{med} = \frac{x(\text{m})}{2} + \frac{x(\text{m}+1)}{2}
\]

Median of a 2D array

```matlab
function \text{med} = \text{medVal}(C) 
[\text{nr}, \text{nc}] = \text{size}(C); 
\text{x} = \text{zeros}(1, \text{nr} \times \text{nc}); 
\text{for} \ r=1:\text{nr} 
\quad \text{x}(\text{r} \times (\text{r}-1) + 1:	ext{r} \times \text{nc}) = C(\text{r}, :) ; 
\end{\text{for}} 
\quad \% \text{Compute median of x and assign to med} 
\% ...
```

See \text{medVal.m}

Back to filtering...

```
for \text{i}=1:\text{m} 
\quad \text{for} \ j=1:\text{n} 
\quad \quad \text{Compute new gray value for pixel (i,j)} 
\quad \text{end} 
\text{end}
```

When window is inside...

```
\text{New gray value for pixel (7,4) =} 
\text{medVal( A(6:8,3:5) )}
```

When window is partly outside...

```
\text{New gray value for pixel (7,1) =} 
\text{medVal( A(6:8,1:2) )}
```

```
function \text{B} = \text{medFilter}(\text{A}, \text{r}) 
\% \text{B from A via median filtering} 
\% with radius \text{r} neighborhoods. 
[\text{m}, \text{n}] = \text{size}(\text{A}); 
\text{B} = \text{uint8}(\text{zeros}(\text{m}, \text{n})); 
\text{for} \ \text{i}=1:\text{m} 
\quad \text{for} \ \text{j}=1:\text{n} 
\quad \quad \text{C = pixel (i,j) neighborhood} 
\quad \quad \text{B}(\text{i}, \text{j}) = \text{medVal}(\text{C}); 
\quad \text{end} 
\text{end}
```
The Pixel (i,j) Neighborhood

\[ i_{\text{Min}} = \max(1, i-r) \]
\[ i_{\text{Max}} = \min(m, i+r) \]
\[ j_{\text{Min}} = \max(1, j-r) \]
\[ j_{\text{Max}} = \min(n, j+r) \]

\[ C = A(i_{\text{Min}}:i_{\text{Max}}, j_{\text{Min}}:j_{\text{Max}}) \]

Mean Filter with radius 3

\[ B = \text{medianFilter}(A, 3) \]

Mean filter fails because the mean does not capture representative values.

\[
\begin{array}{cccccccc}
150 & 149 & 152 & 153 & 152 & 155 \\
151 & 150 & 153 & 154 & 153 & 156 \\
153 & 2 & 3 & 156 & 155 & 158 \\
154 & 2 & 1 & 157 & 156 & 159 \\
156 & 154 & 158 & 159 & 158 & 161 \\
157 & 156 & 159 & 160 & 159 & 162 \\
\end{array}
\]

\[
\begin{array}{cc}
85 & 86 \\
87 & 88 \\
\end{array}
\]

Mean-filtered values

Finding Edges

Near an edge, grayness values change abruptly

\[
\begin{array}{cccccccc}
200 & 200 & 200 & 200 & 200 & 200 \\
200 & 200 & 200 & 200 & 200 & 100 \\
200 & 200 & 200 & 100 & 100 & 100 \\
200 & 200 & 100 & 100 & 100 & 100 \\
200 & 100 & 100 & 100 & 100 & 100 \\
\end{array}
\]

What is an edge?
General plan for showing the edges in an image

- Identify the “edge pixels”
- Highlight the edge pixels
  - make edge pixels white; make everything else black

The Rate-of-Change-Array

Suppose \( A \) is an image array with integer values between 0 and 255.

Let \( B(i,j) \) be the maximum difference between and its eight neighbors.

So \( B(i,j) \) is the maximum value in

\[
A(\max(1,i-1):\min(m,i+1),... \max(1,j-1):\min(n,j+1)) - A(i,j)
\]

Neighborhood of \((i,j)\)

The Rate-of-Change-Array

Suppose \( A \) is an image array with integer values between 0 and 255.

Let \( B(i,j) \) be the maximum difference between and its eight neighbors.

So \( B(i,j) \) is the maximum value in

\[
A(\max(1,i-1):\min(m,i+1),... \max(1,j-1):\min(n,j+1)) - A(i,j)
\]

Neighborhood of \((i,j)\)

The Rate-of-Change-Array

Suppose \( A \) is an image array with integer values between 0 and 255.

Let \( B(i,j) \) be the maximum difference between and its eight neighbors.

So \( B(i,j) \) is the maximum value in

\[
A(\max(1,i-1):\min(m,i+1),... \max(1,j-1):\min(n,j+1)) - A(i,j)
\]

Neighborhood of \((i,j)\)

The Rate-of-Change-Array

Suppose \( A \) is an image array with integer values between 0 and 255.

Let \( B(i,j) \) be the maximum difference between and its eight neighbors.

So \( B(i,j) \) is the maximum value in

\[
A(\max(1,i-1):\min(m,i+1),... \max(1,j-1):\min(n,j+1)) - A(i,j)
\]

Neighborhood of \((i,j)\)

The Rate-of-Change-Array

Suppose \( A \) is an image array with integer values between 0 and 255.

Let \( B(i,j) \) be the maximum difference between and its eight neighbors.

So \( B(i,j) \) is the maximum value in

\[
A(\max(1,i-1):\min(m,i+1),... \max(1,j-1):\min(n,j+1)) - A(i,j)
\]

Neighborhood of \((i,j)\)

The Rate-of-Change-Array

Suppose \( A \) is an image array with integer values between 0 and 255.

Let \( B(i,j) \) be the maximum difference between and its eight neighbors.

So \( B(i,j) \) is the maximum value in

\[
A(\max(1,i-1):\min(m,i+1),... \max(1,j-1):\min(n,j+1)) - A(i,j)
\]

Neighborhood of \((i,j)\)

The Rate-of-Change-Array

Suppose \( A \) is an image array with integer values between 0 and 255.

Let \( B(i,j) \) be the maximum difference between and its eight neighbors.

So \( B(i,j) \) is the maximum value in

\[
A(\max(1,i-1):\min(m,i+1),... \max(1,j-1):\min(n,j+1)) - A(i,j)
\]

Neighborhood of \((i,j)\)

The Rate-of-Change-Array

Suppose \( A \) is an image array with integer values between 0 and 255.

Let \( B(i,j) \) be the maximum difference between and its eight neighbors.

So \( B(i,j) \) is the maximum value in

\[
A(\max(1,i-1):\min(m,i+1),... \max(1,j-1):\min(n,j+1)) - A(i,j)
\]

Neighborhood of \((i,j)\)

The Rate-of-Change-Array

Suppose \( A \) is an image array with integer values between 0 and 255.

Let \( B(i,j) \) be the maximum difference between and its eight neighbors.

So \( B(i,j) \) is the maximum value in

\[
A(\max(1,i-1):\min(m,i+1),... \max(1,j-1):\min(n,j+1)) - A(i,j)
\]

Neighborhood of \((i,j)\)

The Rate-of-Change-Array

Suppose \( A \) is an image array with integer values between 0 and 255.

Let \( B(i,j) \) be the maximum difference between and its eight neighbors.

So \( B(i,j) \) is the maximum value in

\[
A(\max(1,i-1):\min(m,i+1),... \max(1,j-1):\min(n,j+1)) - A(i,j)
\]

Neighborhood of \((i,j)\)

The Rate-of-Change-Array

Suppose \( A \) is an image array with integer values between 0 and 255.

Let \( B(i,j) \) be the maximum difference between and its eight neighbors.

So \( B(i,j) \) is the maximum value in

\[
A(\max(1,i-1):\min(m,i+1),... \max(1,j-1):\min(n,j+1)) - A(i,j)
\]

Neighborhood of \((i,j)\)

The Rate-of-Change-Array

Suppose \( A \) is an image array with integer values between 0 and 255.

Let \( B(i,j) \) be the maximum difference between and its eight neighbors.

So \( B(i,j) \) is the maximum value in

\[
A(\max(1,i-1):\min(m,i+1),... \max(1,j-1):\min(n,j+1)) - A(i,j)
\]

Neighborhood of \((i,j)\)

The Rate-of-Change-Array

Suppose \( A \) is an image array with integer values between 0 and 255.

Let \( B(i,j) \) be the maximum difference between and its eight neighbors.

So \( B(i,j) \) is the maximum value in

\[
A(\max(1,i-1):\min(m,i+1),... \max(1,j-1):\min(n,j+1)) - A(i,j)
\]

Neighborhood of \((i,j)\)

The Rate-of-Change-Array

Suppose \( A \) is an image array with integer values between 0 and 255.

Let \( B(i,j) \) be the maximum difference between and its eight neighbors.

So \( B(i,j) \) is the maximum value in

\[
A(\max(1,i-1):\min(m,i+1),... \max(1,j-1):\min(n,j+1)) - A(i,j)
\]
“Edge pixels” are now identified; display them with maximum brightness (255)

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>90</td>
<td>90</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>90</td>
<td>90</td>
<td>90</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>90</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>90</td>
</tr>
</tbody>
</table>

if \(B(i,j) > \tau\)

\[B(i,j) = 255;\]

End

function Edges(jpgIn,jpgOut,\(\tau\))

% jpgOut is the “edge diagram” of image jpgIn.
% At each pixel, if rate-of-change > \(\tau\)
% then the pixel is considered to be on an edge.
A = rgb2gray(imread(jpgIn));
[m,n] = size(A);
B = uint8(zeros(m,n));
for i = 1:m
    for j = 1:n
        Neighbors = A(max(1,i-1):min(i+1,m), max(1,j-1):min(j+1,n));
        B(i,j)=max(max(abs(double(Neighbors)-
                        double(A(i,j)))));
        if B(i,j) > \(\tau\)
            B(i,j) = 255;
        end
    end
end
imwrite(B,jpgOut,'jpg')

Edge finding: Effect of edge threshold, \(\tau\)

\(\tau\)

\(\tau\)