

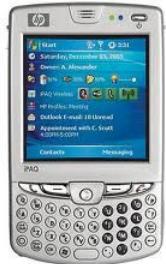
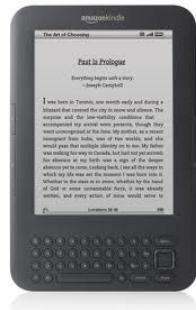
# **CSCE 448/748 - Computational Photography**

---

**Image retargeting**

**Nima Kalantari**

# Display Devices



# Simple Media Retargeting Operators

Letterboxing



# Simple Media Retargeting Operators

- Optimal Cropping Window



- For videos: “Pan and scan”

Still done manually in the movie industry



Liu and Gleicher, Video Retargeting: Automating Pan and Scan (2006)

# Simple Media Retargeting Operators



# Image Retargeting

- **Problem statement:**
  - Input Image  $I$   $n \times m$ , and new size  $n' \times m'$
  - Output Image  $I'$  of size  $n' \times m'$  which will be “good representative” of the original image  $I$
- **To date, no agreed definition, or measure, as to what a good representative is in this context!**

# Example

Scaling



# Content-aware Retargeting Operators

Content-aware



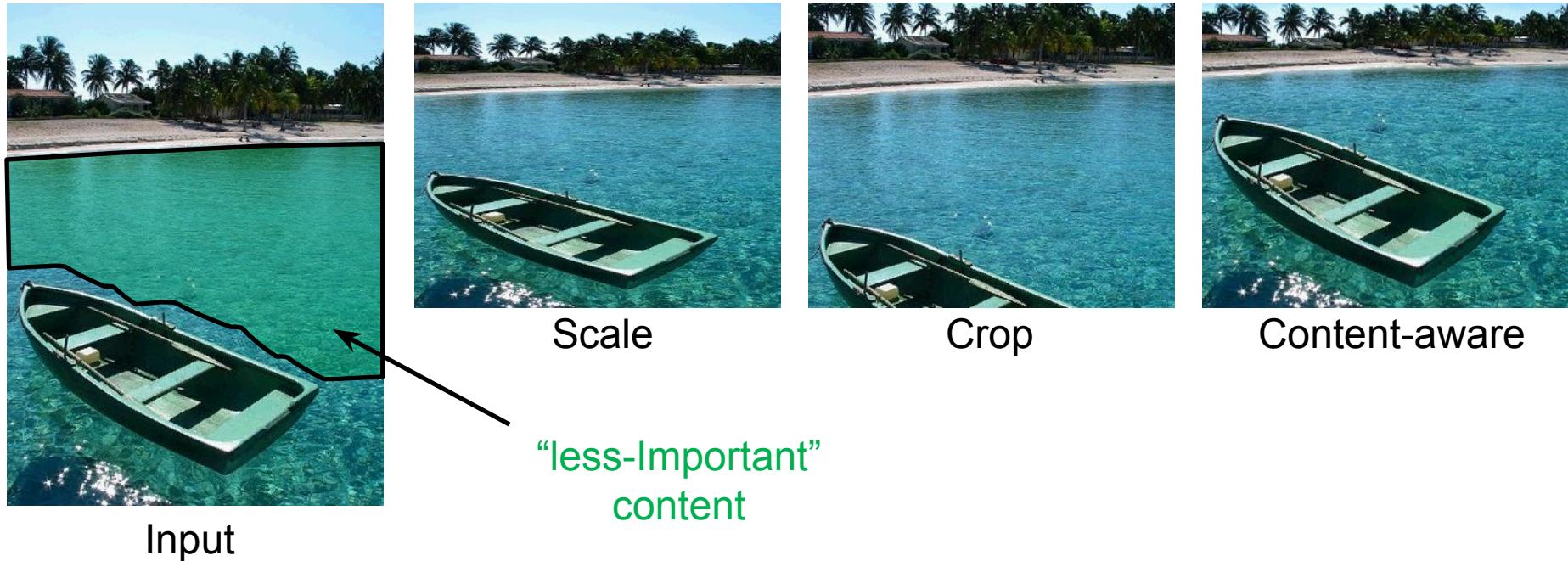
“Important”  
content



Content-oblivious



# Content-aware Retargeting



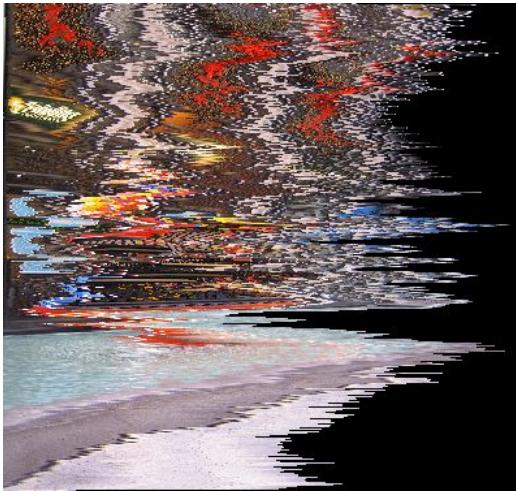
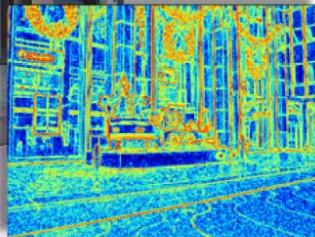
# Seam Carving

- Assume  $n \times m \square n \times m'$ ,  $m' < m$  (summarization)
- Basic Idea: remove unimportant pixels from the image
  - Unimportant = pixels with less “energy”

$$E_1(\mathbf{I}) = \left| \frac{\partial}{\partial x} \mathbf{I} \right| + \left| \frac{\partial}{\partial y} \mathbf{I} \right|.$$

- Intuition for gradient-based energy:
  - Preserve strong contours
  - Human vision more sensitive to edges – so try remove content from smoother areas
  - Simple, enough for producing some nice results
  - See their paper for more measures they have used

# Pixel Removal



Optimal



Least-energy pixels  
(per row)



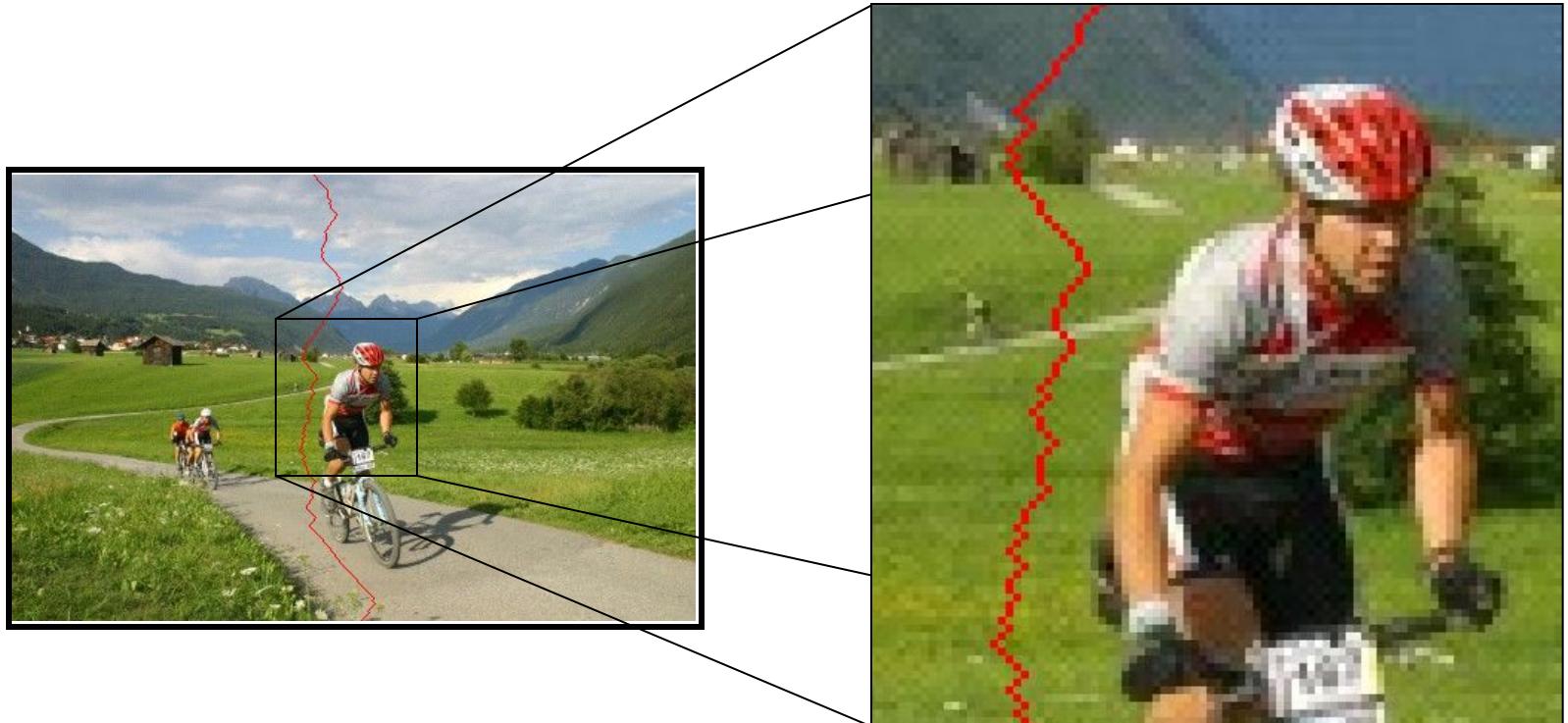
Least-energy columns

# A Seam

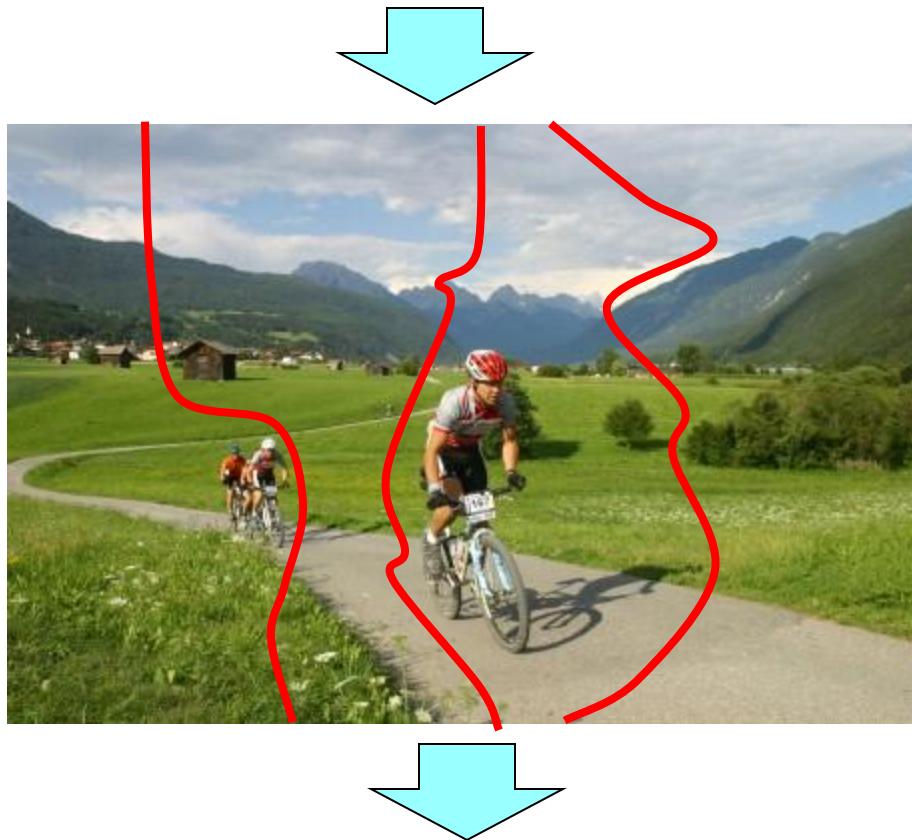
- A connected path of pixels from top to bottom (or left to right). Exactly one in each row

$$\mathbf{s}^x = \{s_i^x\}_{i=1}^n = \{(x(i), i)\}_{i=1}^n, \text{ s.t. } \forall i, |x(i) - x(i-1)| \leq 1$$

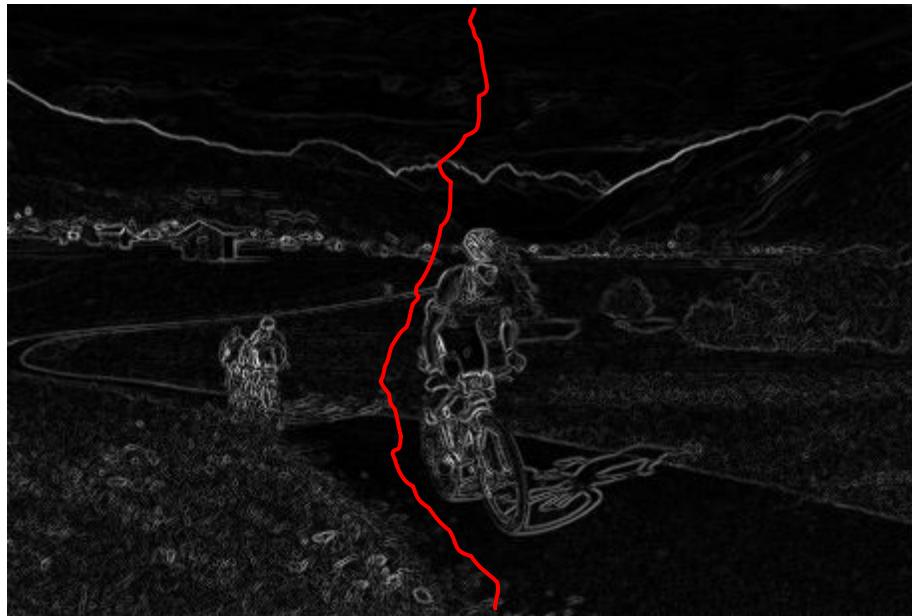
$$\mathbf{s}^y = \{s_j^y\}_{j=1}^m = \{(j, y(j))\}_{j=1}^m, \text{ s.t. } \forall j, |y(j) - y(j-1)| \leq 1$$



# Finding the Seam?



# The Optimal Seam



$$E(\mathbf{I}) = \left| \frac{\partial}{\partial x} \mathbf{I} \right| + \left| \frac{\partial}{\partial y} \mathbf{I} \right| \quad \rightarrow \quad s^* = \underset{s}{\operatorname{argmin}} E(s)$$

# The Optimal Seam

- **The recursion relation**

$$\mathbf{M}(i, j) = E(i, j) + \min(\mathbf{M}(i - 1, j - 1), \mathbf{M}(i - 1, j), \mathbf{M}(i - 1, j + 1))$$

- **Can be solved efficiently using dynamic programming in  $O(s \cdot n \cdot m)$**   
(s=3 in the original algorithm)

# Dynamic Programming

- **Invariant property:**

- $M(i,j)$  = minimal cost of a seam going through  $(i,j)$  (satisfying the seam properties)

5	8	12	3
9	2	3	9
7	3	4	2
6	5	7	8

A 4x4 grid of numbers representing costs for a seam. The grid is as follows:

5	8	12	3
9	2	3	9
7	3	4	2
6	5	7	8

Red arrows point to the value '2' in the second row, second column, indicating it is the current minimum cost for that position in the dynamic programming process.

# Dynamic Programming

$$\mathbf{M}(i, j) = E(i, j) + \min(\mathbf{M}(i - 1, j - 1), \mathbf{M}(i - 1, j), \mathbf{M}(i - 1, j + 1))$$

5	8	12	3
9	<b>2+5</b>	3	9
7	3	4	2
6	5	7	8

A 4x4 grid of numbers. Red arrows point from the top row's values 5, 8, 12, 3 to the second row's value 2+5. The value 2+5 is bolded.

# Dynamic Programming

$$\mathbf{M}(i, j) = E(i, j) + \min(\mathbf{M}(i - 1, j - 1), \mathbf{M}(i - 1, j), \mathbf{M}(i - 1, j + 1))$$

5	8	12	3
9	7	3+3	9
7	3	4	2
6	5	7	8

A 4x4 grid representing a dynamic programming table. The values are: Row 1: 5, 8, 12, 3; Row 2: 9, 7, 3+3, 9; Row 3: 7, 3, 4, 2; Row 4: 6, 5, 7, 8. Red arrows point to the value 3+3 in the second column of the second row, indicating it is the sum of the values from the first column of the second row (9) and the third column of the first row (3).

# Dynamic Programming

$$\mathbf{M}(i, j) = E(i, j) + \min(\mathbf{M}(i - 1, j - 1), \mathbf{M}(i - 1, j), \mathbf{M}(i - 1, j + 1))$$

5	8	12	3
9	7	6	12
14	9	10	8
15	14	15	8+8

# Searching for Minimum

- Backtrack (can store choices along the path, but do not have to)

5	8	12	3
9	7	6	12
14	9	10	8
15	14	15	16

↑

# Backtracking the Seam

5	8	12	3
9	7	6	12
14	9	10	8
15	14	15	16

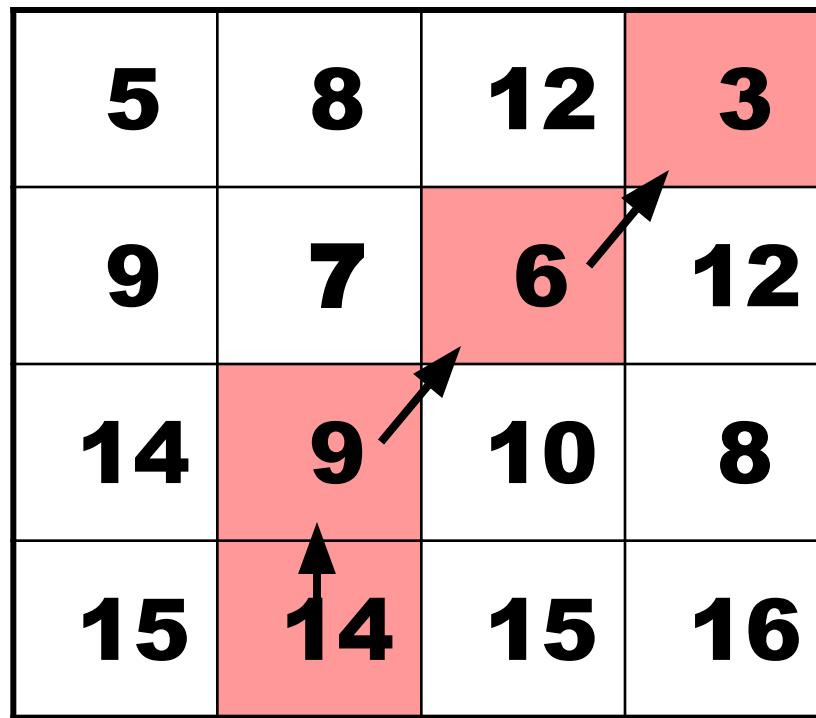
The diagram shows a 4x4 grid of numbers. The top row contains 5, 8, 12, 3. The second row contains 9, 7, 6, 12. The third row contains 14, 9, 10, 8. The bottom row contains 15, 14, 15, 16. The bottom-left cell (14) is highlighted in red. An arrow points upwards from the bottom-left cell to the cell above it (9).

# Backtracking the Seam

5	8	12	3
9	7	6	12
14	9	10	8
15	14	15	16

A 4x4 grid of numbers. The numbers are: Row 1: 5, 8, 12, 3; Row 2: 9, 7, 6, 12; Row 3: 14, 9, 10, 8; Row 4: 15, 14, 15, 16. The cell (3, 2) containing the value 9 is highlighted in red. An arrow points from the cell (3, 3) containing the value 10 to the cell (3, 2). Another arrow points from the cell (4, 2) containing the value 14 up to the cell (3, 2).

# Backtracking the Seam



# The Seam-Carving Algorithm

**SEAM-CARVING(im,m') // size(im) = nxm**

- 1. Do (m-m') times**
  - 2.1. E  $\square$  Compute energy map on im
  - 2.2. s  $\square$  Find optimal seam in E
  - 2.3. im  $\square$  Remove s from im
- 2. Return im**

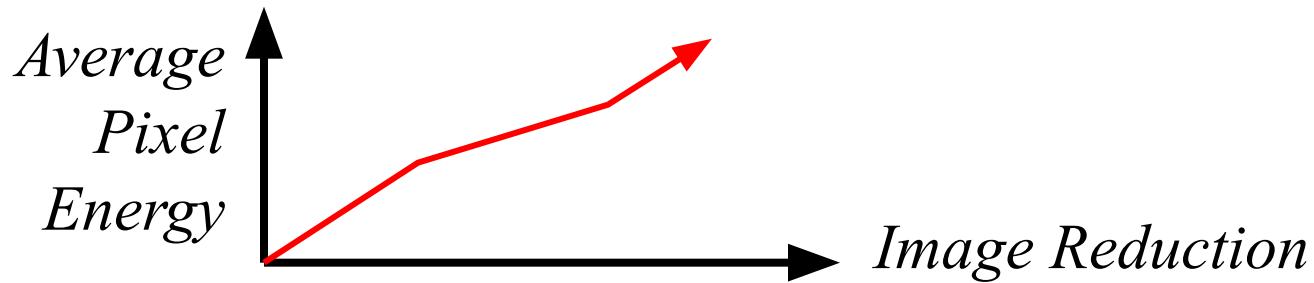
- For vertical resize: transpose the image
- Running time:
  - 2.1  $O(mn)$
  - 2.2  $O(mn)$
  - 2.3  $O(mn)$   
 $\square O(dmn)$   $d=m-m'$

# Preserved Energy

While resizing: remove **as many** low energy pixels and **as few** high energy pixels!

If we measure the average energy of pixels in the image after applying a resizing operator...

...the average should increase!



# Preserved Energy



Average  
Pixel  
Energy

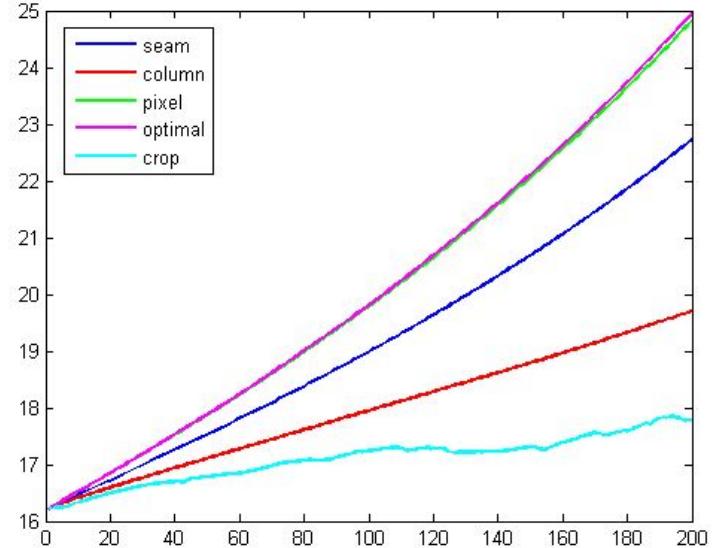


Image Reduction →



crop



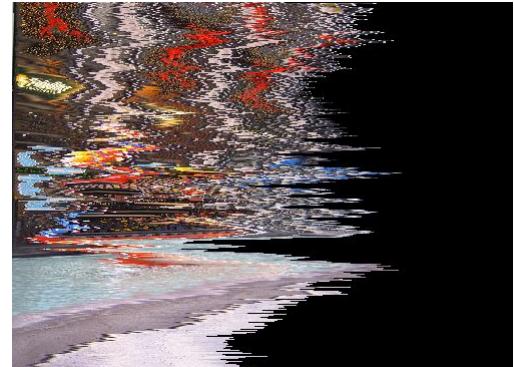
column



seam



pixel



optimal

# Changing Aspect Ratio



# Changing Aspect Ratio



*Original*



*Seam Carving*



*Scaling*

# Changing Aspect Ratio



**Original**



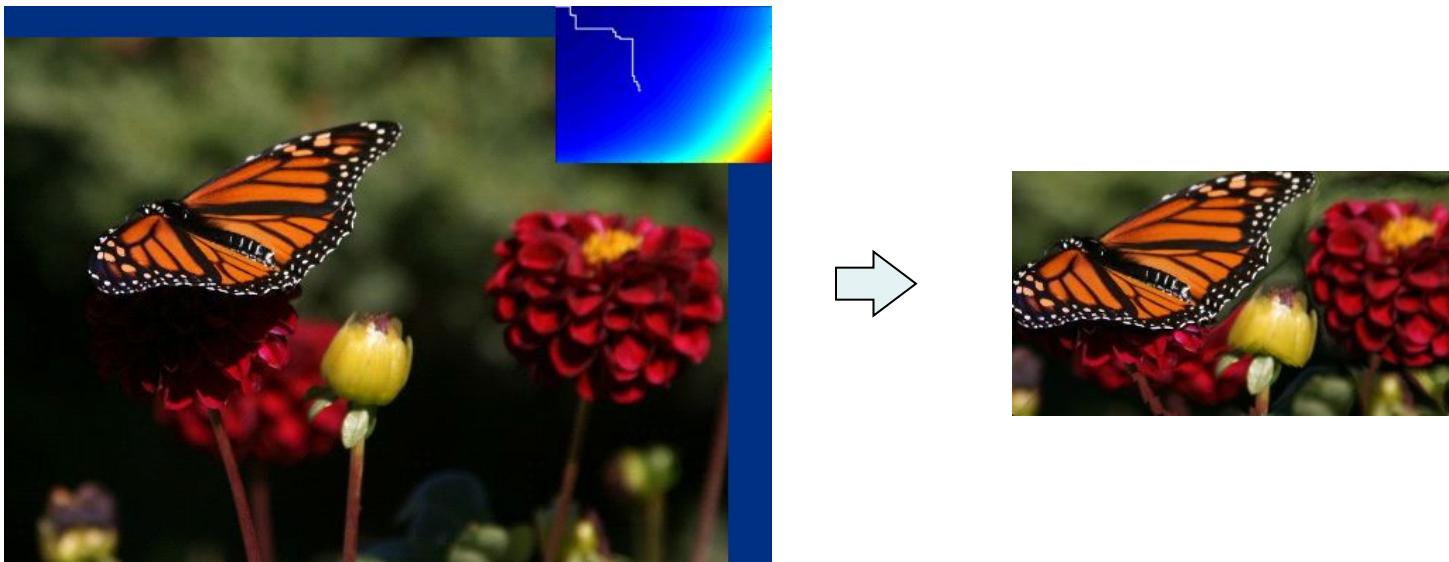
**Retarget**



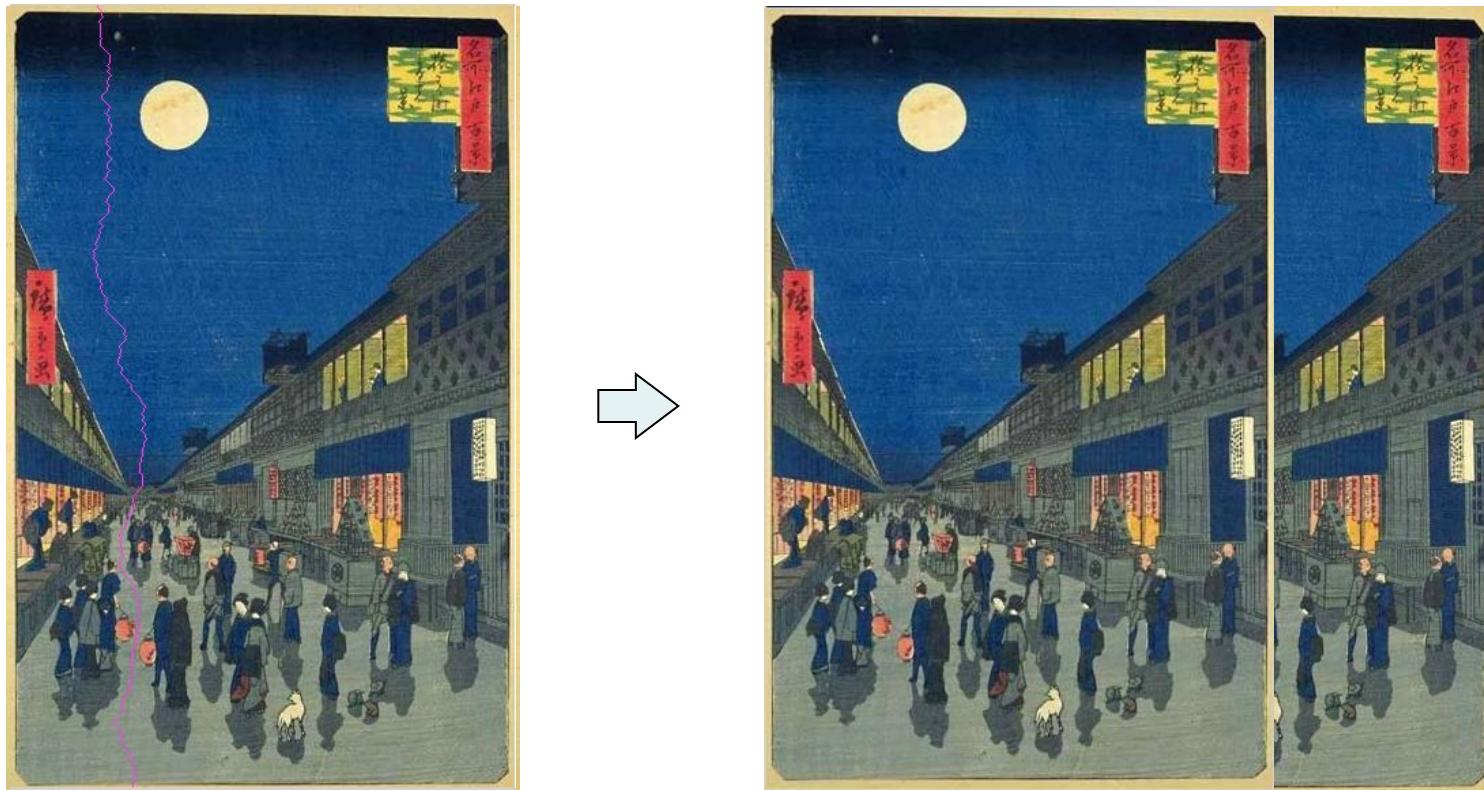
*Scaling*

# Both Dimensions?

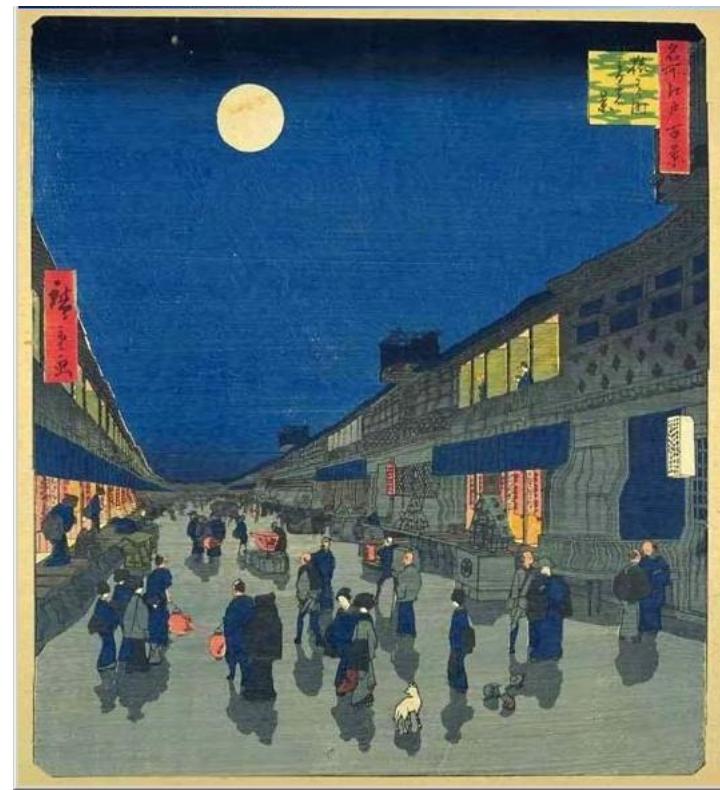
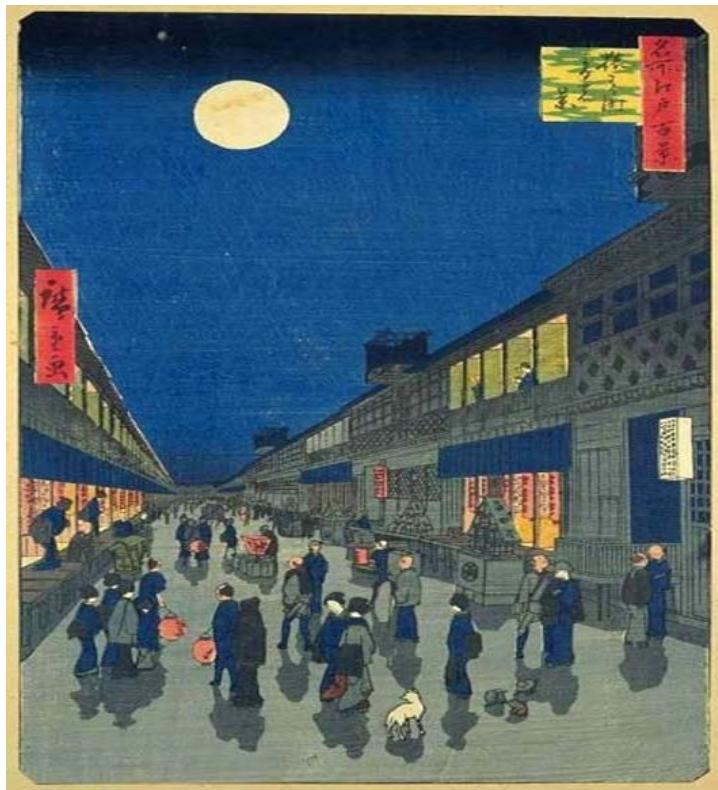
- $m \times n \square m' \times n'$
- Remove horizontal seam first?
- Remove vertical seams first?
- Alternate between the two?
- The optimal order can be found!  $\square$  Dynamic Prog (again)



# Image Expansion (Synthesis)

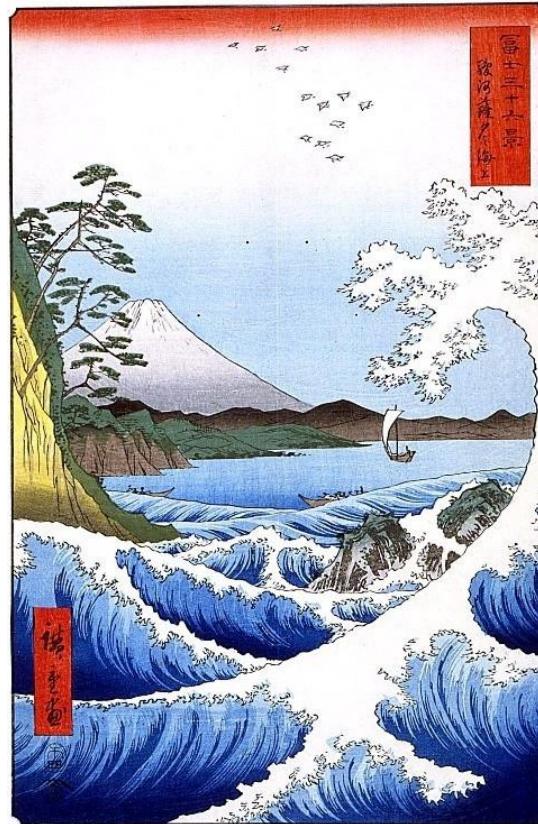


# Image Expansion – take 2



Scaling

# Enlarged or Reduced?



# Combined Insert and Remove



*Insert & remove seams*



*Scaling*

# With face detector



# With User Constraints

Original



Cropped



Scaled



Seam Carved



Face



Face and Flower

# Object Removal



# Object Removal



Input

Retargeted

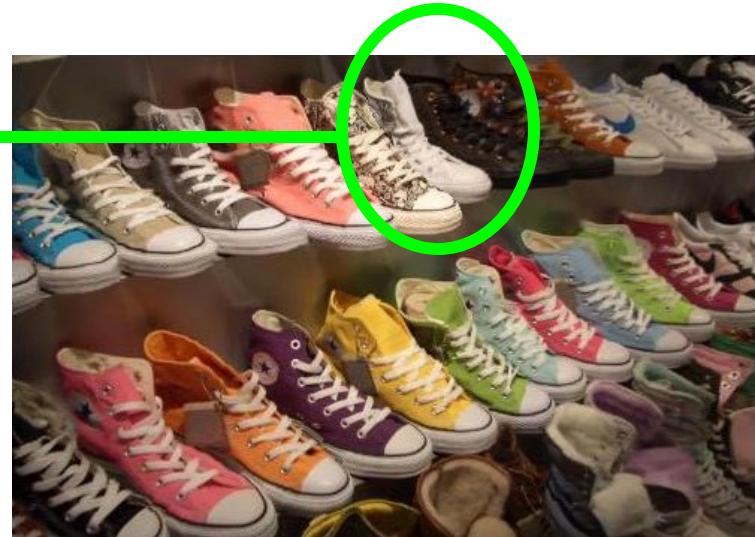
Pigeon Removed

Girl Removed

# Find the Missing Shoe!



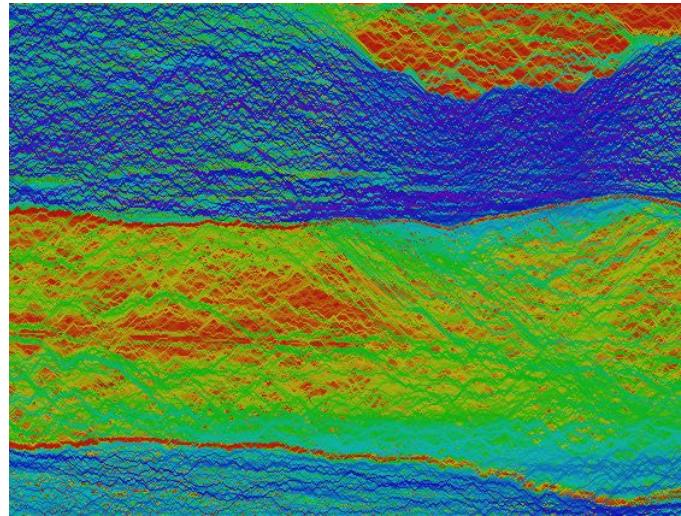
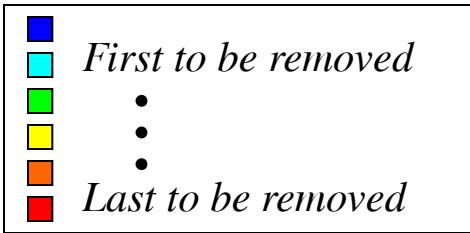
# Solution



# Multi-Size Images

- We can create a new representation of an image that will allow adapting it to different sizes!
  1. Precompute all seams once
  2. Realtime resizing / transmit with content

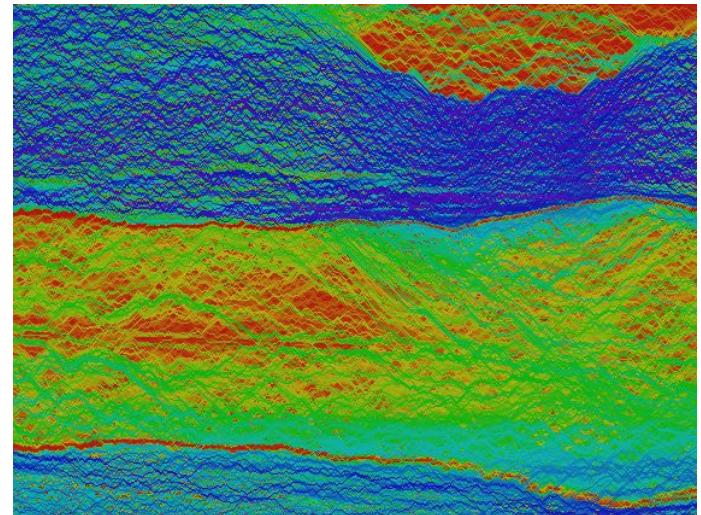
# Multi-Size Images



# Multi-Size Image Representation

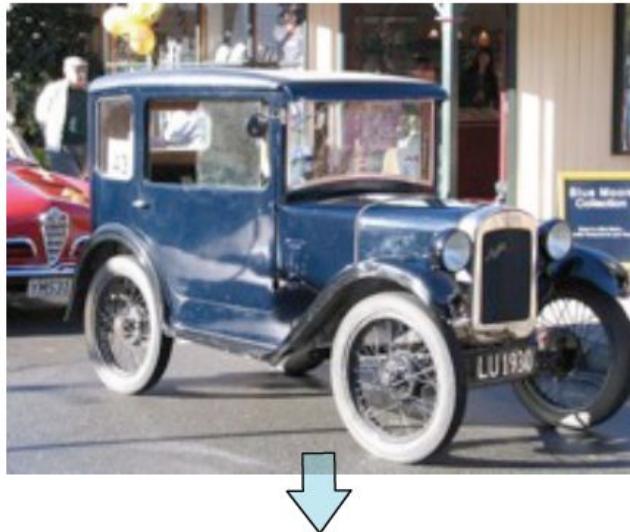


+



# Limitations

Content



Structure

