# Image and Video Processing
# Laboratory 5
# Images Retargeting

Prof. Touradj EBRAHIMI

6 - 7 December 2017

**Reports submission – !!! read carefully !!!** Students will have to produce and submit a report two weeks after completion of each laboratory session. An electronic version of the report is submitted via Moodle's interface for uploading assignments. There will be a hard deadline for each submission. Late submissions will have to be justified and explained to the responsible assistant by email. A structure of the report is explained below. In addition to the report, the source code produced should also be submitted in electronic form. The report and the source code should be submitted in **ONE ZIP** file using Moodle platform.

**Structure of the reports and m-files**

- the name of the final ZIP file will be: `lab_`*number-of-lab_your-surname*`.zip`
- the final ZIP file will contain
  - final m-file called `run_lab_`*number-of-lab_your-surname*`.m`
  - all created m-files and functions which are necessary to obtain the results
  - all source images and data needed for successful running of the final m-file
  - an electronic version of your report in **PDF** format
- the structure of the final m-file
  - the final m-file will contain all necessary commands and functions, by running this m-file one must get all results[1]
  - submitted m-files will be commented
  - each figure will be properly titled
  - the final m-file will be divided into cells[2] according to the example bellow
- the parts of program codes should not be included in the final report
- description of the results should be a part of the final report
- don't forget to answer all the questions in your report
- check whether your final m-file can be launched without problems - only then submit your report

[1] of course it is upon you whether you want to include everything in the final m-file, or create different functions which you will call in the final m-file

[2] those who do not know how to use the cells in Matlab to create the program code more transparent and easier to read, they should look at `http://www.mathworks.com/demos/matlab/developing-code-rapidly-with-cells-matlab-video-tutorial.html`

**An example of final m-file**

```
% Lab (number of lab) - your name and date

%% Exercise (number of exercise) - "Name of exercise"

% your own program code with your coments
a = imread('picture.tif');
% all figures will be properly titled
figure('name','Name Of The Figure')
imshow(a,[]);

%% Exercise (number of exercise) - "Name of exercise"

% your own program code with your coments
[output1, output2, ..., outputN] = function_name (input1, input2, ..., inputN);
```

# 0 Introduction

The huge variety of devices enabling the visualization of multimedia content, especially images, forces content providers to implement solutions resizing any images when targeting a specific resolution of image. Content retargeting is the operation of taking the end-user scenario (e.g. end-user profile and device capabilities) into account for multimedia content delivery[1]. The study [2] compared the retargeting solutions proposed by researchers for the last decades. It has been emphasized that multi-operators are a reference in terms of content retargeting. This Lab aims at the implementation of a simple version of a multi-operator when reducing images size.

**Note:** All the exercises have to be applied on both test images (*kodim*03*.png* and *kodim*23*.png*). If not mentioned otherwise, the targeted image size is the DCIF format (384 x 528).

# 1 Cropping

Cropping is the simplest retargeting operator. It consists in preserving only a portion of the source content. Given an image $I$ ($x$ x $y$), the cropped image $I_c$ ($x_c$ x $y_c$ with $x_c < x$ and $y_c < y$) results from cutting some of the image from the left, right, top and bottom or any combination thereof.

1. Manual Cropping

   For both test images, select the window of size 384 x 528 which, according to your perception, will keep the important information of the source image.

   Report the source coordinates of the up-left pixel of $I_c$. Describe the process you followed to select $I_c$. What energy function could correspond to your perceptual process?

2. Automatic Cropping

   In any automatic cropping operation, an energy function is defined to indicate the importance of a window in the image. This energy function is applied on all the possible windows of size $x_c$ x $y_c$ in $I$. The retained window $I_c$ is the window having the highest energy.

   Implement the automatic cropping operator using as energy function the sum over a window of the normalized image gradient:

   $$e(I) = |\frac{\partial I}{\partial x}| + |\frac{\partial I}{\partial y}| \tag{1}$$

   The gradient is applied on the luminance channel of the image $I$. You can use the function rgb2gray of matlab. You can filter an image in both direction using the Sobel filter (See Lab 3) or the function imgradientxy of matlab. The cropped image $I_c$ should be in the DCIF format (384 x 528).

   Comment on the results obtained. Comment on the used energy function.

## 2 Resizing

A resizing operation consists in reducing the size of the image by applying filtering. In this exercise, the image source $I$ ($x$ x $y$) is resized to half its size in both directions: $I_r$ ($x_r$ x $y_r$) with $x_r = x/2$ and $y_r = y/2$.

1. Sub-sampling

   Remove one pixels out of two, in both horizontal and vertical dimensions, from the source image (See Lab 1).

2. Bi-linear interpolation

   The previous sub-sampling preserves only a fourth of the source image information. Using a bi-linear interpolation allows to substitute four pixels with an interpolation of thereof. The figure 1 presents the process of bilinear interpolation: the pixels of $I_r$ result from two linear interpolations along the horizontal dimension, followed by a linear vertical interpolation.

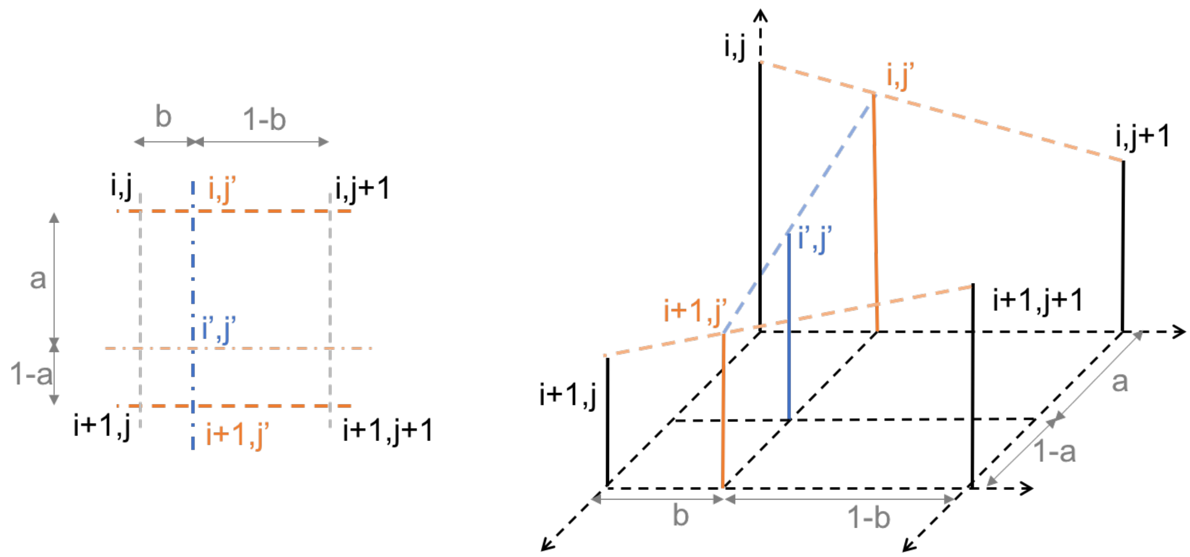


Figure 1: Bi-linear interpolation scheme

   Given the image $I$, let $I_r(i', j')$ be the pixel to interpolate from $I(i, j)$, $I(i + 1, j)$, $I(i, j + 1)$ and $I(i + 1, j + 1)$, write the interpolation equations leading to the bi-linear interpolation $I_r(i', j')$:

   - interpolate $I_r(i, j')$ as a function of $b$, $I(i, j)$ and $I(i, j + 1)$ and $I_r(i + 1, j')$ as a function of $b$, $I(i + 1, j)$ and $I(i + 1, j + 1)$,
   - interpolate $I_r(i', j')$ as a function of $a$, $I_r(i, j')$ and $I_r(i + 1, j')$ and
   - write the equation to interpolate $I(i', j')$ as a function of $a$, $b$, $I(i, j)$, $I(i, j + 1)$, $I(i + 1, j)$ and $I(i + 1, j + 1)$.

   What are the values of a and b in our context? Implement the bi-linear interpolation of $I$.

Do NOT use the imresize function from matlab.

Comment on your results and compare the two approaches.

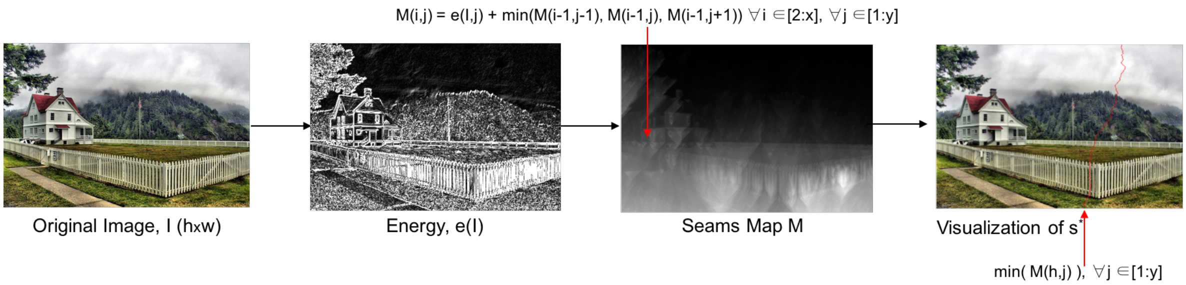


Figure 2: Horizontal seam carving block diagram

# 3 Seam Carving

Seam carving is a content-aware resizing operator, described in [3]. It consists, in the context of image size reduction, in removing seams of least energy from the source picture until reaching the targeted size. A seam is an optimal 8-connected path of pixels on an image which follows one direction only (horizontally from top to bottom or vertically from left to right). The optimality of the seam is defined by an energy function.

The process followed by this operator is described below.

- Compute the energy of the image
- Compute the map of seams
- Find the seam of least energy $s^*$
- Remove $s^*$ from $I$
- Visualize the removed seam $s^*$ in $I$

The block diagram presented in Figure 2 illustrates the seam carving algorithm process.

Given an source image $I$ ($x$ x $y$), to obtain the seam-carved image $I_{sc}$ ($x_{sc}$ x $y_{sc}$), we have to apply the seam-carving algorithm $x - x_{sc}$ times horizontally and $y - y_{sc}$ times vertically. The direction of the seam to be removed is selected based on a cost function.

## 3.1 Vertical Seam Carving

The implementation of seam carving with only vertical seams is detailed hereinafter:

1. Compute the energy of the image

   The energy function used for this operator is the sum of the normalized vertical and horizontal gradients of the image I, defined in equation 1.

2. Compute the seams map

   Let $I$ be an image of size $x$ x $y$, a vertical seam is defined to be:

   $$s^v = \{s_i^v\}_{i=1}^x = \{(i, f(i))_{i=1}^x, s.t. \forall i, |f(i) - f(i-1)| \leqslant 1\} \quad (2)$$

   where $f$ is a mapping $f : [1, ..., x] \mapsto [1, ..., y]$. That is, a vertical seam is an 8-connected path of pixels in the image from top to bottom, containing one, and only one, pixel in each row of the image (see Figure 2).

   Given an energy function e, the cost of a seam is defined as $E(s) = E(I_s) = \sum_{i=1}^{x} e(I(s_i))$, with $I_s = \{I(s_i)_{i=1}^x\} == \{I(f(i), i))_{i=1}^x\}$. The optimal seam $s^*$ minimizes the seam cost: $s^* = \underset{s}{min} E(s)$.

The optimal seam $s^*$ can be found by dynamic programming when creating a seams map. The seams map $M$ is constructed from the second row until the last row. It contains the cumulative minimum energy for all possible connected seams for each entry (i,j):

$$M(i,j) = e(i,j) + min(M(i-1,j-1), M(i-1,j), M(i-1,j+1)) \tag{3}$$

3. Find the seam of least energy

   The values contained in the last row of M represent the cost of a seam starting from the pixel $(x, j)$. $s^*$ is the seam starting with the pixel $I(x, j^*)$, with $j^*$ the element of the last row of M with the least energy.

   To recover the entire seam $s^*$, backtrack from $j^*$ in M to find the path of the optimal seam.

4. Remove the seam $s^*$ from $I$

   Similarly to row or column removal, removing the pixels of a seam from an image has only a local effect: pixels of the image positioned after the seam are shifted left (or up if horizontal seam) to compensate for the missing path.

5. Visualize the removed seam $s^*$ in $I$

   It is of importance to visualize which information have been deleted from the source picture. However, the indexes of pixels are modified at each iteration of the seam carving process. In order to visualized which information has been removed from the source content, it is necessary to store the source indexes of the kept pixels.

   Also, should the seam carving algorithm have been applied n times, n seams should be observed on the visualization.

   **Hint:** You can create a structure containing the indexes of the seams removed.

Implement the vertical seam carving. Observe and analyze the algorithm behavior when resizing $I$ to $x$ x $(y-50)$ and $x$ x $(y-200)$ images.

### 3.2 Horizontal Seam Carving

Although the implementation of horizontal seam-carving is very similar to vertical seam-carving, slight modifications have to be applied on the seam definition and the seams map computation. For instance, if y is a mapping $g : [1, ..., y] \mapsto [1, ..., x]$, then a horizontal seam is:

$$s^h = \{s^h_j\}^y_{j=1} = \{(g(j), j)^y_{j=1}, s.t. \forall j, |g(j) - g(j-1)| \leqslant 1\} \tag{4}$$

**Hint:** Horizontal seam carving can be implemented as a vertical seam carving applied to the transposed image of $I$.

Implement the horizontal seam carving. Observe and analyze the algorithm behavior when resizing $I$ to $(x-50)$ x $y$ and $(x-200)$ x $y$ images.

### 3.3 Vertical and Horizontal Seam Carving

In order to optimally resize an image while removing only the least important information, horizontal and vertical seam-carving operations are interleaved. The cost function, establishing which of the optimal vertical seam $s^*_v$ or the optimal horizontal seam $s^*_h$ will be the optimal seam $s^*$, is defined as the comparison between the energy of seams $s^*_v$ and $s^*_h$. This means

$$s^* = \begin{cases} s^*_v & if \quad j^*_v \leqslant j^*_h \\ s^*_v & otherwise \end{cases} \tag{5}$$

Implement the vertical and horizontal seam carving. Observe and analyze the algorithm behavior when resizing $I$ to the DCIF format and 100 x 150 targeted size.

# 4 Multi-operator

The authors of [4] proposed to combine several resizing operator in a multi-operator in order to optimize resizing processes. To do so, several constraints have to be taken into account:

- Selection of operators

  The operators to be used as a part of the multi-operator can be selected based on efficiency or complexity. This process is complex when considering the numerous available resizing solutions. The first multi-operator, to be implemented in this Lab, consists in the combination of three operators: cropping, bi-cubic resizing and seam carving.

  Operators part of the multi-operator should be discrete and separable in dimension: a two dimensional resizing can be treated as a sequence of horizontal and vertical size reduction operations. That is, when an operator is applied, only one dimension is modified. The selection of three operators results in six sub-operators for the multi-operator. Thus, the sub-operators part of the multi-operator to be implemented are:

  - Horizontal Seam Carving (HSC),
  - Vertical Seam Carving (VSC),
  - Horizontal Cropping (HC),
  - Vertical Cropping (VC),
  - Horizontal Resizing (HR) and
  - Vertical Resizing (VR).

  **Hint:** You can use the function imresize of matlab for the bi-cubic resizing operator.

- Selection of number of iterations (application of sub-operators)

  The aim of a multi-operator is not to select one operator and apply it to the picture. A sequence of sub-operators will be applied to the picture to obtained $I_{mo}$. A multi-operator iteration is defined as modifying the size of the picture, by atomically discarding information in one dimension. Let $x_{mo}$ and $y_{mo}$ be the target size of the image $I$ ($x$ x $y$), an iteration consists in resizing $I_{it}$ ($x_{it}$ x $y_{it}$) to $I_{it+1}$ ($x_{it+1}$ x $y_{it+1}$) with

  $$\begin{cases} x_{it+1} = x_{it} - k, y_{it+1} = y_{it} & if \quad \text{horizontal retargeting} \quad \text{with } 0 < k < x - x_{mo} \\ y_{it+1} = y_{it} - k, x_{it+1} = x_{it} & otherwise \quad \text{with } 0 < k < y - y_{mo} \end{cases} \tag{6}$$

  In most cases $k = 1$. However, to fasten the processing, we set k as a multiple of $x - x_{mo}$ and $y - y_{mo}$. There would then be $(x - x_{mo} + y - y_{mo})/k$ iterations in the multi-operator operation.

- Definition of a sub-operators order

  It would be possible to define an order in the use of the sub-operators. However, this constraint would limit the optimization of the resizing operation.

- Selection of a discriminative metric

  As the order of sub-operators in not pre-defined, we need a metric or cost function which will enable the comparison of the operators in order to discriminate which one to use. Full-reference metrics (comparing source and impaired images) cannot be used in this context due to the different size of source and retargeted images. The aim of the multi-operator, as any retargeting operation, is to preserve as much of the source image information as possible. Thus, metrics evaluating the entropy or the amount of edges in a picture are suitable. Several metrics have been proposed:

  - Entropy (E) of the luminance channel of image $I_{it}$
    $E(I) = -\sum(p * log2(p))$, with $p$, the normalized histogram of the luminance channel of $I$.
  - Sum of normalized Gradient (Grad) of the luminance channel of image $I_{it}$
    Sum, over all pixels, of the energy of the luminance channel of $I$. The energy function is presented in equation 1.

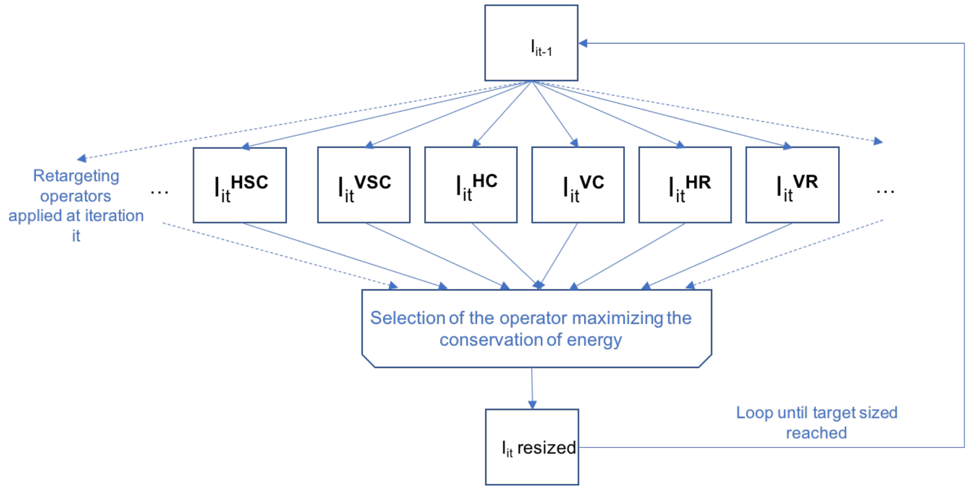


Figure 3: Multi-operator block diagram

- Entropy Difference (ED) between $I_{it}$ and $I$
  $ED(I_{it}) = E(I) - E(I_{it})$
- Sum of normalized Gradient Difference (GradD) between $I_{it}$ and $I$
  $GradD(I_{it}) = Grad(I) - Grad(I_{it})$

You can use the function entropy and imgradientxy of matlab to implement those metrics.

For your information, you already have implemented a multi-operator. The combination of vertical and horizontal seam carving is a multi-operator having as energy function the cost of a seam. This metric is not applicable with cropping or resizing as no seam can be implemented in those operators.

## 4.1 Greedy algorithm

In this multi-operator variant, the selection of the operator is made at every iteration. The figure 3 presents the block diagram of the multi-operator. At every iteration, the results of the sub-operators are compared using the selected metric. If the metric selected is the entropy or the sum of normalized gradients, $I_{i+1}$ is the sub-operator image having the maximum energy. If the metric selected is the difference of entropy or the difference of the sum of normalized gradients, then $I_{i+1}$ is the result image having the minimum energy. In the first case, the approach consists in maximizing the amount of information in $I_{i+1}$, in the latter, the approach is to guarantee a similar amount of energy in $I_{i+1}$ and $I$.

Implement this variant of the multi-operator when targeting the DCIF format, for the four energy functions, for $k = \{4, 8\}$. Comment on your results (e.g the use of different energy function and the order of operators).

## 4.2 Optimization

Another way to optimize the multi-operator is to compute every possible combination of the operators for all iterations as described in [4]. The selected retargeted image $I_{mo}$ is selected based on the energy function (maximum if E and Grad, minimum if ED and GradD). The order of the operators is backtracked from $I_{mo}$. For this multi-operator variant, the function OptimizedMultiOp is provided.

Comment on the results of this variant, compare it with the previous one.

# References


[1] A. El Saddik and M. S. Hossain, "Multimedia content repurposing," in *Encyclopedia of Multimedia*. Springer, 2006, pp. 494–500.

[2] M. Rubinstein, D. Gutierrez, O. Sorkine, and A. Shamir, "A comparative study of image retargeting," in *ACM transactions on graphics (TOG)*, vol. 29, no. 6. ACM, 2010, p. 160.

[3] S. Avidan and A. Shamir, "Seam carving for content-aware image resizing," in *ACM Transactions on graphics (TOG)*, vol. 26, no. 3. ACM, 2007, p. 10.

[4] M. Rubinstein, A. Shamir, and S. Avidan, "Multi-operator media retargeting," in *ACM Transactions on graphics (TOG)*, vol. 28, no. 3. ACM, 2009, p. 23.