Computational Photography **2014/2015** **P3**

Seam Carving




Material:

[1] Original paper: http://www.win.tue.nl/~wstahw/2IV05/seamcarving.pdf

[2] Demo: http://swieskowski.net/carve/

Abstract: Effective resizing of images should not only use geometric constraints, but consider the image content as well. We present a simple image operator called *seam carving* that supports content-aware image resizing for both reduction and expansion. A seam is an optimal 8-connected path of pixels on a *single* image from top to bottom, or left to right, where optimality is defined by an image energy function. By repeatedly carving out or inserting seams in one direction we can change the aspect ratio of an image. By applying these operators in both directions we can retarget the image to a new size. The selection and order of seams protect the content of the image, as defined by the energy function. Seam carving can also be used for image content enhancement and object removal. We support various visual saliency measures for defining the energy of an image, and can also include user input to guide the process. By storing the order of seams in an image we create *multi-size* images, that are able to continuously change in real time to fit a given size.

En esta práctica vamos a implementar **SEAM CARVING**, un algoritmo para reducir automáticamente el tamaño de una imagen, teniendo en cuenta su contenido visual.

Los pasos principales que se tienen que implementar en esta práctica son 3:

1. Seam carving para la **REDUCCIÓN** automática del contenido de la imagen. Dada una imagen y un número de iteraciones, aplicar el algoritmo para reducir el número de filas y/o columnas de la imagen. Utilizar la imagen "countryside.jpg".

   Seam carving para la **ELIMINACIÓN** de un objeto de la imagen. Dada una imagen, seleccionar manualmente el contorno de un objeto que queremos eliminar y aplicar el algoritmo para crear una nueva imagen sin ese objeto. Utilizar la imagen "agbar.png" o bien una **fotografía vuestra**.

   El fichero "interactive_only.py" permite dibujar un contorno con el botón izquierdo del mouse. El contorno se cierra clicando con el botón derecho. Atención, no hace falta clicar en el punto de origen, el código conecta el último punto con el primero. El resultado sera un mask del contorno dibujado.

   Si se usa Spyder, es necesario presionando F6 cambiar de "execute in current Python interpreter" cambiar en "Execute in a new dedicated interpreter

2. Seam carving para la **SINTESIS** de imágenes. Dada una imagen, crear una versión de tamaño **SUPERIOR** aplicando el algoritmo. Podéis utilizar la imagen countryside.jpg o bien una **fotografía vuestra**.

Proponemos repartir estas tres tareas en tres semanas de trabajo, aunque cada grupo puede organizarse como quiera, ya que la entrega es única y al final de la tercera semana.

Detalles de implementación

1. El algoritmo de seam carving se basa en una función de energía calculada a partir de la imagen. En [1] los autores proponen dos funciones de energía que se pueden utilizar. Para la nuestra implementación, la función (1) del artículo es suficiente, donde I es la imagen:

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

2. Una vez calculada la energía, se tiene que calcular la matriz M de energía mínima acumulada:

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

Los índices *(i,j)* indican las **FILAS** y **COLUMNAS** de la matriz, respectivamente.

3. Dada la matriz M, calcular el SEAM, o sea el recorrido desde ABAJO hacía ARRIBA (o DERECHA > IZQUIERDA) que indique los pixeles que se tienen que borrar para reducir el contenido de la imagen de manera "inteligente". El calculo del seam se hace a través del proceso de ***backtracking*** (pag. 46 clase 6).

4. Para implementar la eliminación de un objeto, podéis utilizar el codigo del fichero "interactive_only.py" Para asignar un valor bajo de energía al objeto que se quiere eliminar, un coeficiente de valor -100 aplicado a la función E ya es suficiente.

6. Para calcular las derivadas de la imagen según las direcciones (x,y) podéis utilizar la función gradient() de python.

Presentar los resultados

Entregar todo el código de la practica, y un PDF que explique los resultados en el Campus Virtual **antes de la fecha limite.**

Para entregar y presentar vuestros resultados, es necesario:

- Subir en una carpeta **TODO el código python y las imágenes utilizadas al Campus Virtual** de la UB antes de la fecha límite establecida con el profesor de prácticas.

- Explicar posibles problemas y soluciones. La idea del texto es simplemente añadir informaciones útiles al lector para entender e interpretar el resultado conseguido.