Image inpainting with OpenCV

Image inpainting is a form of image conservation and image restoration, dating back to the 1700s when Pietro Edwards, director of the Restoration of the Public Pictures in Venice, Italy, applied this scientific methodology to restore and conserve famous works.

Technology has advanced image painting significantly, allowing us to:

- Restore old, degraded photos
- Repair photos with missing areas due to damage and aging
- Mask out and remove particular objects from an image (and do so in an aesthetically pleasing way)

There are two OpenCV's inpainting algorithms. From there, we'll implement an inpainting demo using OpenCV's built-in algorithms, and then apply inpainting to a set of images.

OpenCV's inpainting algorithms:

The OpenCV library ships with two inpainting algorithms:

- cv2.INPAINT_TELEA: An image inpainting technique based on the fast-marching method (Telea, 2004)
- cv2.INPAINT_NS: Navier-stokes, Fluid dynamics, and image and video inpainting (Bertalmío et al., 2001)

How does inpainting work with OpenCV?

When applying inpainting with OpenCV, we need to provide two images:

1. The input image we wish to inpaint and restore. Presumably, this image is "damaged" in some manner, and we need to apply inpainting algorithms to fix it.
2. The mask image, which indicates where in the image the damage is. This image should have the same spatial dimensions (width and height) as the input image. Non-zero pixels correspond to areas that should be inpainted (i.e., fixed), while zero pixels are considered "normal" and do not need inpainting.

USAGE:

python opencv_inpainting.py --image examples/example01.png --mask examples/mask01.png

Implementing inpainting with OpenCV and Python

Our script is set up to handle four command line arguments at runtime:

- **--image:** The path to the damaged photograph upon which we'll perform inpainting
- **--mask:** The path to the mask, which corresponds to the damaged areas in the photograph
- **--method:** Either the "telea" or "ns" algorithm choices are valid inpainting methods for OpenCV and this Python script. By default (i.e., if this argument is not provided via the terminal), the Telea et al. method is chosen
- **--radius:** The inpainting radius is set to 3 pixels by default; you can adjust this value to see how it affects the results of image restoration

Inpainting with OpenCV couldn't be any easier — simply call the built-in inpaint function while passing the following parameters:

- **image:** The damaged photograph
- **mask:** The single-channel grayscale mask, which highlights the corresponding damaged areas of the photograph

- **inpaintRadius:** The radius in pixels is the “circular neighborhood of each point inpainted that is considered by the algorithm” (OpenCV docs); in our case, it comes directly from the --radius command line argument
- **flags:** Holds the inpainting method (either cv2.INPAINT_TELEA or cv2.INPAINT_NS)

The return value is the restored photograph (output).

We display three images on-screen:

1. our original damaged photograph
2. our mask which highlights the damaged areas
3. the inpainted (i.e., restored) output photograph

Each of these images will remain on your screen until any key is pressed while one of the GUI windows is in focus.

Sources: https://www.pyimagesearch.com/2020/05/18/image-inpainting-with-opencv-and-python/