

Region Filling and Object Removal by Exemplar-Based Image Inpainting

By
Swagata Naskar
19EC39038

Algorithm :

For a given input image, user first selects the target region Ω to be removed and filled to suit the background. This background (Φ) or $(I - \Omega)$ would be called the source region.

The boundary of the target region is called contour/fill-front and is represented by $\delta\Omega$. As this is a patch based filling process, patch size must be specified. Generally, it is taken as 9×9 . However it is a better idea to take the patch size to be slightly bigger than the largest distinguishable texture element.

As soon as the target region is selected, a mask(binary image) is made in which \forall pixels $p \in \Omega$; 1 is assigned, and the rest of the image, 0. And then a new image is created where target region Ω is white-washed. We fill the color values in this image and display the output.

During the course of algorithm, every pixel in the image will be assigned a **confidence value** which shows how confident we are in the color value of pixel. We also **calculate and assign priority value for patches centered on the pixels along the contour**. The patch with the highest priority value will be filled first. The priority computation is biased toward those patches which:

- (i) are on the continuation of strong edges and
- (ii) are surrounded by high-confidence pixels. Now the following three steps will be looped until the required output is obtained.

1.. Computing patch priorities:

For a patch ψ_p centered on a pixel $p \in \delta\Omega$, the priority $P(p) = C(p)D(p)$. Here $C(p), D(p)$ are called **confidence term**, data term respectively. Confidence of a pixel $c(q)$ is set to be 0 $\forall q \in \Omega$ and 1 otherwise in the beginning. The confidence term $C(p)$, data term $D(p)$ for a patch centered at pixel p are given by:

$$C(p) = \frac{\sum_{q \in \psi_p \cap (I - \Omega)} c(q)}{|\psi_p|}$$
$$D(p) = \frac{|\nabla I_p^\perp \cdot n_p|}{\alpha}$$

Here $|\psi_p|$ is the patch size. Here ∇I_p^1 stands for the **image gradient vector** which gives the isophote direction and intensity at that point. We considered the point in the patch at which there is the highest intensity to represent the patch. \mathbf{n}_p is the unit normal vector at the pixel p . α is the normalization factor.

2.. Filling the patches:

Once all the priorities of the patches on the boundary are computed, we proceed to start filling the patch with the highest priority. As we discussed earlier, we are not going to use diffusion for this purpose. We copy the information directly from the "ideal sample" in the source region. We pick the sample which has the minimum Euclidean distance with respect to the patch that needs to be filled.

It has to be noted that the distance is to be calculated only for the pixels that are already filled in the patch that needs to be filled. Once we found such an exemplar patch in the source region, for each pixel that needs to be filled, we copy the value from its corresponding position inside the exemplar patch.

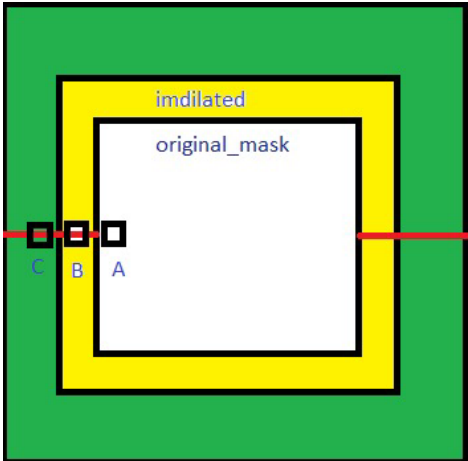
3.. Updating the confidence values:

$$\mathbf{c}(\mathbf{p}) = \mathbf{C}(\mathbf{p}') \forall \mathbf{p} \in \psi_{p'} \cap \Omega$$

Here $\mathbf{c}(\mathbf{p})$ stands for confidence value of an individual pixel whereas $\mathbf{C}(\mathbf{p}')$ stands for the confidence term evaluated for the patch centered at \mathbf{p}' which lies between **0 and 1**. It is to be noted that the confidence values are only being updated for newly filled pixels. Once filled, the confidence value of a pixel freezes.

Thus, as the filling proceeds, confidence values decay denoting that we are less sure of the values that are filled at the center of target region Ω . After this the contour will be updated.

Discussion:

- *Green color and yellow color together are the source region and the line shows the direction of the isophote. For calculating the isophote value and direction of a pixel 'A' on the contour, we will consider a patch of size $n \times n$ centered at the pixel 'A', and the maximum isophote value in that patch is assigned as the isophote value of 'A'. We will not consider the pixels in the yellowed shaded region to avoid the gradient occurring due to color change by our original mask. The imdilated mask size is less than the patch size at pixel A. There could be a steep gradient near the boundary of the target region since we whitewash the target region. So we performed immediate operation on the mask and when calculating the data term of patches, we didn't consider the gradient values at the pixels which are present in the vicinity of the boundary. This improved our results.*
- 
- The diagram shows a green square representing the source region. Inside it is a yellow square representing the imdilated mask. Inside the yellow square is a white square representing the original_mask. A horizontal red line passes through the center of the white square. Three small black squares are marked on this line: one in the green region (labeled 'C'), one in the yellow region (labeled 'B'), and one in the white region (labeled 'A').
- *In the case of Figure 5, we can clearly see that the patches at sharp edges are being filled first. This is because, as the paper already suggested, the confidence term is biased towards sharp edges.*
 - *In figure 5(c), Since the data term is biased towards the patches where the isophote values are higher and places where isophotes flow into, the patches present near the boundaries with high gradient were filled first. But as we reach further into the target region, confidence decays and the patches near other boundaries get filled.*
 - *If we happen to find multiple patches that are equidistant from the patch which is to be filled, we consider the patch with the least gradient value at its center to be the exemplar patch. This can happen especially when images are not natural. Ex: Flags, designs etc. This helped us statistically with propagating linear structures.*

- *First we used uint8 format for the image when we were trying to find the exemplar patch from the source region. But, negative numbers were type-casted as zeros on the LHS. So, we were not able to calculate the distances accurately. We finally debugged and changed the format to double and got the results right.*
- *The paper took into consideration the maximum isophote value within the patch into consideration when calculating the data term (normal. *isophote). Our results were getting noisy when we tried implementing that, but directly computing the data term for all pixels and then taking into consideration the maximum value helped us.*
- *We have attached both the code files considering patch size 9*9*
***Code1 ///** considering max isophote*
***Code2 ///** considering max data values*