Image inpainting and completion

Onofre Martorell Nadal
onofremartorelln@gmail.com

Universitat Oberta de Catalunya

Abstract. This document is a brief explanation of the work done the project of the Module 2 of the master of Computer Vision. The goal of the project was to implement an algorithm of image completion based on different variational and graphical models. The document is structured as follows: first, a short introduction of the problem; later, the steps done so as to solve the problem and at the end, an example of the solution obtained and the conclusions.

Keywords: image completion, image inpainting, image segmentation, image editing

1 Introduction

Every once in a while, we all wish we could erase something from our old photographs. A garbage truck right in the middle of a charming Italian piazza, an ex-boyfriend in a family photo,... Other times, there is simply missing data in some areas of the image. An aged corner of an old photograph, a dead bug on the camera lens or the red parts of the image 1. The way to restore those images is through a technique called image completion. Image completion (also called hole-filling) is the task of filling in or replacing an image region with new image data such that the modification can not be detected. This does not mean that we can recover the information that it was originally there, it means that we can add new information in order that a human eye can not detect that this process has been done in the image.

With that, the aim of this project is to present an algorithm for image completion based on different techniques based on variational and graphical models. In order to have a more clear structure, the document has been divided in four parts: inpainting, segmentation, Poisson editing and completion, which corresponds with sections 2 to 3. Each section contains a brief explanation of the technique used in an step and the way to implement it. After all that, the results are presented using image 1 as an example of the process.

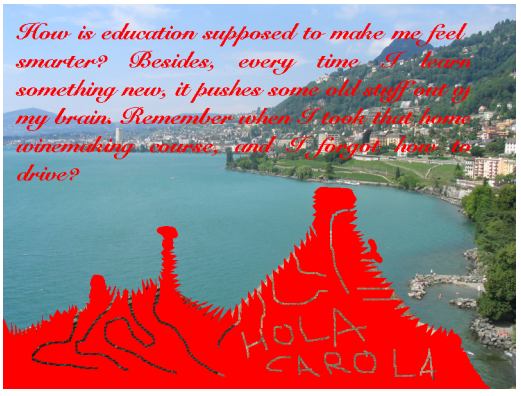


Fig. 1: example image in which we will perform the image completion algorithm.

2 Image inpainting

2.1 Formulation of the model

The first step done is image inpainting, which consists on reconstructing lost parts of the image. Concretely, we will use this technique in order to eliminate the letters from our image. The way to do it is the following: let $f : \Omega \to \mathbb{R}$ be a given grayscale image and let $D \subset \Omega$ be an open set representing the region to be inpainted. With this, the inpainted image u can be find as the solution of the functional

$$\begin{cases} \underset{u \in W^{1,2}(\Omega)}{\arg\min} \displaystyle\int_D |\nabla u(x)|^2 dx, \\ u|_{\partial D} = f \end{cases} \tag{1}$$

The interpretation of this functional is that we want to find the image u in which the change of color between neighbour pixels is small and that is similar to the original image in the boundary of the set.

2.2 Numerical implementation

The minimum of the functional from the equation 1 also satisfies the associated Euler-Lagrange equation

$$\begin{cases} u_{xx} + u_{yy} = 0 \text{ in } D \\ u = f \qquad\quad \text{in } \partial D. \end{cases} \tag{2}$$

Given a pixel (i, j) from the image and belonging to D, this equation can be discretizised as [1] [2]

$$\frac{1}{h_j^2} u_{i,j-1} + \frac{1}{h_i^2} u_{i-1,j} - \left(\frac{2}{h_i^2} + \frac{2}{h_j^2} \right) u_{i,j} + \frac{1}{h_i^2} u_{i+1,j} + \frac{1}{h_j^2} u_{i,j+1} = 0. \tag{3}$$

Moreover, for the boundary of the image we add one row or column to each side of the image and we compute the Neumann boundary conditions. For example, at the east side, we get

$$\frac{u_{i,1} - u_{i,2}}{h_j} = 0 \Longrightarrow u_{i,1} = u_{i,2}. \tag{4}$$

And for the points not belonging to D we have

$$u_{i,j} = f_{i,j}. \tag{5}$$

To sum up, we sort the pixels of the image as

$$x = (u_{1,1}, u_{2,1}, \cdots, u_{i,j}, u_{i+1,j}, \cdots, u_{n_i+2,n_j+2})^T \tag{6}$$

and for each pixel we create its equation (3, 4 or 5) corresponding with the type of pixel it is. Doing that, the inpainted image u can be found as the solution of the linear system

$$Ax = b, \tag{7}$$

where A is the matrix of coefficients and b is the vector of independent terms for each equation. As one have may noticed, we defined f as a grayscale image, but in fact our image is colorscale. The way to apply this algorithm to image 1 is by performing it on each channel of color (RGB). This results can also be improved using another algorithm such the one explained in [3].

3 Image segmentation

Once we have eliminated the letters, the next step is to remove the other red part of the image, and the way to do it will be by copying a zone from another image that fits on our image. This step will be done on section 5, but before that we need to do some steps: as on can see in the image 1, the boundary of the red zone of the bottom is not smooth. Because of that, the next step we will do is to make a segmentation of the image in order to have an smoother boundary. The way to segment the image will be done using a technique developed by Mumford and Shah in [4] and improved by Chan and Vese in [5].

3.1 Formulation of the model

The Chan-Vese segmentation finds the curve C that is the boundary of the segmentation. As it is difficult to manipulate C we will use a function ϕ and the C will be the zero crossing of φ that is

$$C = \{x \in \Omega : \varphi(x) = 0\}. \tag{8}$$

With this the functional that we need to minimize is

$$\begin{aligned} \arg\min_{c_1,c_2,\varphi} \mu \int_\Omega \delta(\varphi(x))|\nabla\varphi(x)|dx + \nu \int_\Omega H(\varphi(x))dx + \lambda_1 \int_\Omega |f(x) - c_1|^2 H(\varphi(x))dx \\ +\lambda_2 \int_\Omega |f(x) - c_2|^2 (1 - H(\varphi(x)))dx, \end{aligned} \tag{9}$$

where H denotes the Heaviside function and δ the Dirac mass, its distributional derivative:

$$H = \begin{cases} 1 & t \geq 0, \\ 0 & t < 0 \end{cases}, \quad \delta(t) = \frac{d}{dt} H(t). \tag{10}$$

Note that we cannot derive $H(t)$. Because of that, in the implementation we take the Heaviside function and the Dirac mass as

$$H_\varepsilon(t) = \frac{1}{2}\left(1 + \frac{2}{\pi}\arctan\left(\frac{t}{\varepsilon}\right)\right), \quad \delta_\varepsilon = \frac{\varepsilon}{\pi(\varepsilon^2) + t^2)}. \tag{11}$$

3.2 Numerical implementation

The functional 9 is minimized as in [6] and the way to do it is the following: at each iteration we do this steps

1. Update c_1 and c_2 as

$$c_1 = \frac{\int_\Omega f(x)H(\varphi(x))dx}{\int_\Omega H(\varphi(x))dx} \text{ and } c_2 = \frac{\int_\Omega f(x)(1 - H(\varphi(x)))dx}{\int_\Omega (1 - H(\varphi(x)))dx}. \tag{12}$$

2. Evolve φ using the semi-implicit Gauss-Seidel method [7]

$$\begin{aligned} \varphi^n_{i,j} = [\varphi^n_{i,j} + dt \cdot \delta_\varepsilon(\varphi^n_{i,j})(A_{i,j}\varphi^n_{i+1,j}) + A_{i-1,j}\varphi^{n+1}_{i-1,j} + B_{i,j}\varphi^n_{i,j+1} + \\ B_{i,j-1}\varphi^{n+1}_{i,j-1} - \nu - \lambda_1(f_{i,j} - c_1)^2 + \lambda_2(_{i,j} - c_2)^2)] \\ /[1 + dt \cdot \delta_\varepsilon(\varphi^n_{i,j})(A_{i,j} + A_{i-1,j} + B_{i,j} + B_{i,j-1})], \end{aligned} \tag{13}$$

   where

$$A_{i,j} = \frac{\mu}{\sqrt{\eta^2 + (\varphi^n_{i+1,j} - \varphi^n_{i,j})^2 + ((\varphi^n_{i,j+1} - \varphi^{n+1}_{i,j-1})/2)^2}} \tag{14}$$

$$B_{i,j} = \frac{\mu}{\sqrt{\eta^2 + ((\varphi^n_{i+1,j} - \varphi^{n+1}_{i-1,j})/2)^2 + (\varphi^n_{i,j} - \varphi^n_{i+1,j})^2}} \tag{15}$$

3. If we have reached the maximum number of iterations or the difference $\max(|\varphi^{n+1} - \varphi|)$ is lower than a given tolerance, we stop the algorithm.

As one can notice, the functional 9 has many different parameters. In order to clarify the use of each of them and the value used in the implementation, the table 1 has been created.

4 Image Poisson Editing

As we said at the beginning, we would like to copy some information from another image to our, but if we do that without any tuning, the new information would seem to be a fake image. In order to avoid this, we will adapt the new information to the existing one using a process called Poisson Editing. In the paper [8], there are described two ways for doing the image poisson editing. In this case, we will use the first one, which is called importing gradients.

Parameter	Use	Value in implementation
μ	Weighs the length of the curve C	1
ν	Weighs the area inside the curve C	0
λ_1	Weighs the discrepancy between piecewise model u and the input image inside C	10^{-3}
λ_2	Weighs the discrepancy between piecewise model u and the input image outside C	10^{-3}
ε	Used in the regularization of the Heaviside function in equation 10	1
tolerance	Parameter used to determine when the algorithm stops	10^{-4}
dt	Step of the gradient-descent algorithm	$(10^{-1})/\mu$
η	Positive value to avoid the denominators of equations 14 and 15 being zero	1

Table 1: all the meaning and values of the parameters used in the implementation of the Chan-Vese segmentation of section 3.

4.1 Formulation of the model

The importing gradients method works as follows: let f^* be the destination image, f the source image that contains the region we want to clone, $S \subset \mathbb{R}^2$ the domain of the image, $\Omega \subset S$ a closed subset which will be cloned from f to f^* and $\overrightarrow{v}$ the guidance field of vectors defined over Ω. To solve the problem we have used the next functional over each channel of the image:

$$\min_f \int_\Omega |\nabla f - \overrightarrow{v}|^2, \tag{16}$$

with $f_{|\delta\Omega} = f^*_{|\delta\Omega}$.

4.2 Numerical implementation

The solution of 16 is the unique solution of the following Poisson equation with Dirichlet boundary conditions:

$$\begin{cases} \Delta f = \operatorname{div} \overrightarrow{v} & \text{on} \quad \Omega \\ f = f^* & \text{in } \partial\Omega \end{cases} \tag{17}$$

As we said, we are using importing gradients, which means that the guidance field $\overrightarrow{v}$ is a gradient field taken directly from a source image. Denoting by g the source image, this is $\overrightarrow{v} = \nabla g$. Replacing this in equation 17, we get

$$\begin{cases} \Delta f = \Delta g & \text{on} \quad \Omega \\ f = f^* & \text{in } \partial\Omega \end{cases} \tag{18}$$

In section 2 we have implemented a numerical method to solve the equation 2 which is a particular case of equation 18. What we have done is the following: we have taken the code of the previous week and we have changed it in order to detect if there is a variable containing the discretization of Δg for each point of the image. If this variable exists, it computes $\Delta f = \Delta g$; otherwise it computes $\Delta f = 0$.

5 Image completion

Now we are at the last step of the process in order to complete image completion. This last part of the project is based on [9] and these are the steps done to complete the process:

1. From a huge database of image, find the most similar images using GIST descriptors.
2. From every found image, find the most similar region to the hole of our image.

3. Authomatically fine tune of the editing mask. This step consists on taking the segmentation obtained in section 3 and tuning it using a graphical model. Concretely, we will use graph cuts, and the cost function is

$$C(L) = \sum_{p} C_d(p, L(p)) + \sum_{p,q} C_i(p, q, L(p), L(q)) \tag{19}$$

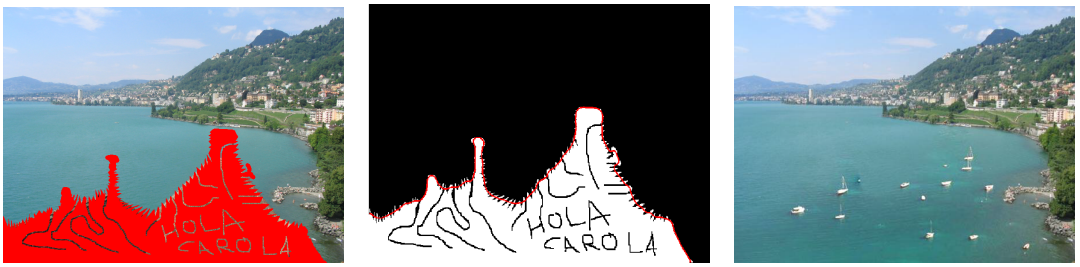
where $L(p)$ is the label of the pixel p, which can be *exists* or *patch*, C_d are the unary potentials and C_i are the binary potentials.
4. Copy the region to our image and perform Poisson Editing correction from section 4.
5. As we will have many results, choose by visual inspection the best one.

6 Results and conclusions

Figure 2 shows all the steps performed by the algorithm. As it can be observed, the result is quite good and the final image doesn't seem to be tuned by a computer. The implementation of this algorithm is not very difficult, in fact, the most difficult part is to understand all the formulas and equations that appear during the process, but they are understood, it's easy to implement.

References

1. Joe D Hoffman and Steven Frankel. *Numerical methods for engineers and scientists*. CRC press, 2001.
2. Note that besides the ones indicated in this document, there are many books that explain how to obtain this equation.
3. Xavier Bresson and Tony F Chan. Fast dual minimization of the vectorial total variation norm and applications to color image processing. *Inverse problems and imaging*, 2(4):455–484, 2008.
4. David Mumford and Jayant Shah. Optimal approximations by piecewise smooth functions and associated variational problems. *Communications on pure and applied mathematics*, 42(5):577–685, 1989.
5. Tony F Chan and Luminita A Vese. Active contours without edges. *IEEE Transactions on image processing*, 10(2):266–277, 2001.
6. Pascal Getreuer. Chan-vese segmentation. *Image Processing On Line*, 2:214–224, 2012.
7. Gilles Aubert and Luminita Vese. A variational method in image recovery. *SIAM Journal on Numerical Analysis*, 34(5):1948–1979, 1997.
8. Patrick Pérez, Michel Gangnet, and Andrew Blake. Poisson image editing. In *ACM Transactions on Graphics (TOG)*, volume 22, pages 313–318. ACM, 2003.
9. James Hays and Alexei A Efros. Scene completion using millions of photographs. *Communications of the ACM*, 51(10):87–94, 2008.



(a) Inpainting of the red letters in image 1.

(b) Segmentation of the red zone remaining in 2a.

(c) One possible final result of the image completion.

Fig. 2: steps of the algorithm presented on the document. Image 2a corresponds to section 2, image 2b corresponds to section 3 and image 2c corresponds to section 5.