Project 2: Image Inpainting
Total Variation Inpainting

MTH-238 Spring 2022
Professor Jue Wang

Khai Dong

May 2022

1 Abstract

Total Variation was introduced for image denoising and reconstruction in 1992 by Rudin, Osher, and Fatemi [2]. Currently, It is widely used in both the industry and research projects.
In this project, we are going to explore the application of Total Variation in Image Inpainting and how the math and assumptions that enable Image Inpainting. It also explores how Total Variation is lacking in image inpainting and provides insights on how it can be improved.
Along with the theoretical knowledge about Image Inpainting, this project also provides an implementation of such a method in Python (Appendix A).
The project's video link is in Appendix B. The video is a summarized presentation of this project's report over Zoom.
The images used are taken and generated by the writer, so there is no need to give credit.

2 Introduction

Image Inpainting is the task of reconstructing "damaged" regions on an image. In this process, the "damaged" regions are specified, and the new contents of these regions are inferred from their surrounding contents in a way that the new contents "blend" with the image. Image inpainting is traditionally used to restore damaged artworks both physically and digitally. Moreover, it is also used to solve disocclusion (to estimate the background behind an obscuring foreground object). By extension, by replacing the object with the approximated background, the removal of the obscuring foreground object is possible (e.g., foreground objects can range from a trashcan, scars, blemishes, etc., to overly noisy regions of an image). In today's photography which concentrates on the after editing of the images, image inpainting is widely used to remove

imperfections from the images.
Here is an example of the powerful application of image inpainting in photography where the obstructing building and wire in the top left corner are inpainted from the image (Figure 1).
As such, image inpainting is available in many popular photo editing software (e.g., Adobe Photoshop). This project is going to investigate the method of TV (Total Variation) Inpainting.

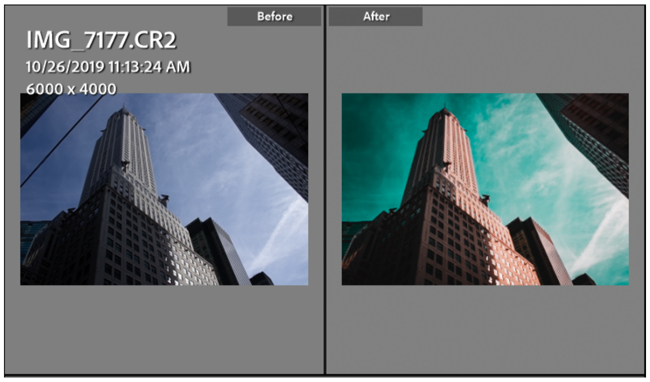


Figure 1. Before and After of An Image of Empire State Building (Taken and Edited By Dong, Khai)

In this project, for the sake of simplicity, I will only work with grayscale images.

3 Body

3.1 Total Variation

Total Variation was introduced for image denoising and reconstruction in 1992 by Rudin, Osher, and Fatemi [2]. Since we are considering a gray-scale image, we consider each pixel of the image to be $x \in \mathbb{R}$. As such, the total variation of image X can be defined as follows:

$$TV(X) = \sum_{i,j \in \mathcal{N}} \|x_i - x_j\|_2 \tag{1}$$

where $\mathcal{N}$ is the pixel neighborhood (usually adjacent pixels both horizontally and vertically) and $\|\cdot\|_2$ is the ℓ_2 norm. Due to time constraints, this project is only going to use ℓ_2 norm in the TV Inpainting.
Let the given grayscale image X be defined by the function $f : \Omega \to \mathbb{R}$ where Ω is the set of image's pixels. We have that, locally, the gradient ∇f is the vector between 2 pixel neighborhood, i and j. This means the magnitude of this vector is $|\nabla f| = \|x_i - x_j\|_2$. Therefore, we have that the total variation can be rewritten in the form of a functional

$$TV(f) = \int_{\Omega} |\nabla f| dx \tag{2}$$

3.2 Total Variation Inpainting

To use calculus of variation on the image inpainting problem, we define the components of the problem as follows:

- Let the given grayscale image X be defined by the function $f : \Omega \to \mathbb{R}$ where Ω is the set of image pixels.
- Let $D \subset \Omega$ be the region that needed to be inpainted. We suppose that $f(\Omega \setminus D)$ is known and $f(D)$ is "damaged".
- Let the inpainted image be defined by the function $u : \Omega \to \mathbb{R}$. In the approximation, we would want $f = u$ in $\Omega \setminus D$ since $\Omega \setminus D$ is the non-damaged region.

By defining the problem this way, if u is smooth, TV is going to be the functional

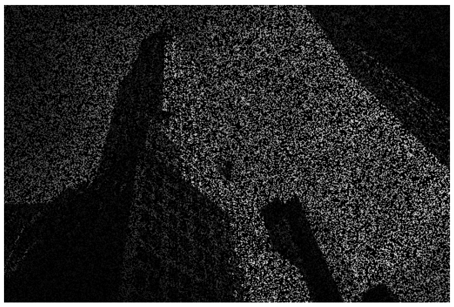
$$TV(u) = \|u\|_{TV(\Omega)} = \int_{\Omega} |\nabla u| dx \tag{3}$$

where $|\nabla u|$ is the magnitude of the gradient ∇u. We can then approximate u which yield the lowest total variation by finding the extremal of the functional $TV(u)$

$$\arg \min_{u \in BV(\Omega)} \|u\|_{TV(\Omega)} \tag{4}$$

3.3 Demonstration

Firstly, we have a set of damaged grayscale image and the mask of the damaged regions as follows


2a. Damaged

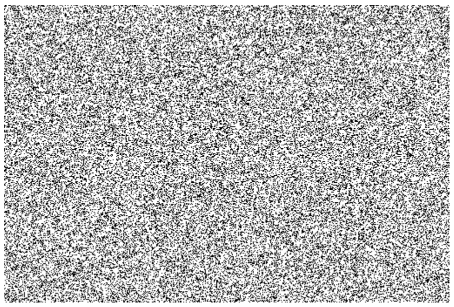

2b. Mask

Figure 2. Input Image and "Damaged" Region Mask

The black pixels in the mask represent the damaged pixels in the image.
Here, we want to recover the "damaged" pixels whose regions are represented as the binary mask to obtain an approximation of the original image (Figure 4b).
To achieve this we are going to minimize the Total Variation of the image. In

order to solve the minimization problem, I created a Python script which used the package cvxpy which supports solving the functional minimization using SCS. The code and the photos used are available in Appendix A.
After executing the Python script, I got the following output (Figure 4a).

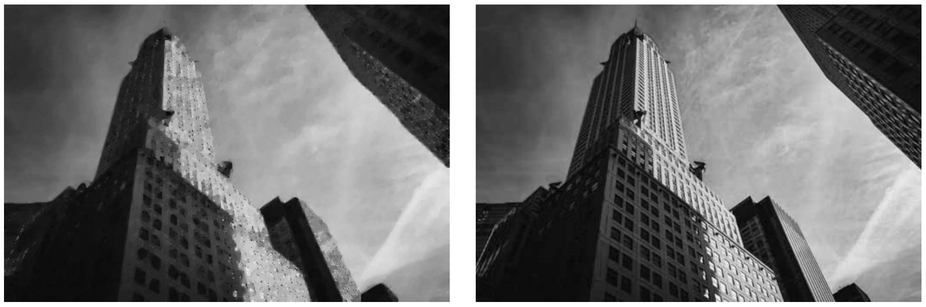

4a. Inpainted 4b. Original

Figure 4. Inpainted vs. Original

We can see that compared to the original, the inpainted is overly smooth. This is because we assume that u is smooth whereas, in natural images, u is cluttered and only locally smooth. Hence, approximating u this way will have a strong denoising effect on u which caused the new D to be too smooth.

3.4 A More Sophisticated Method

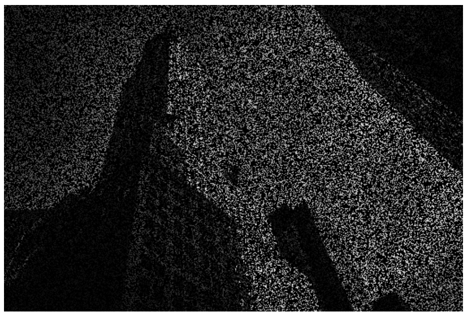
Getreuer (2012) proposed the minimization of the following functional [1]

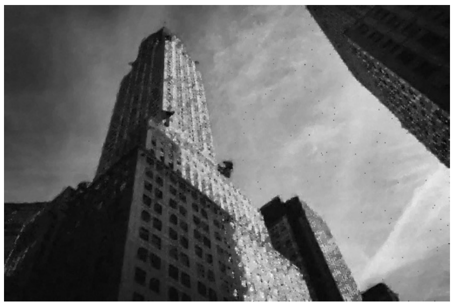
$$TV(u) = \|u\|_{TV(\Omega)} + \frac{\lambda}{2}\int_{\Omega\setminus D}(f(x) - u(x))^2 dx \tag{5}$$

where the term $\frac{\lambda}{2}\int_{\Omega\setminus D}(f(x) - u(x))^2 dx$ will penalize the inpainted u for having too much variation with f in their respective pixel x in known region $\Omega \setminus D$. We would want λ to a large positive number to minimize the denoising effect since our objective is to inpaint the damaged region. This ensures $u(D)$ will match with $f(\Omega \setminus D)$ without $u(D)$ being too smooth compared to f. Therefore, we now have to solve the following minimization problem [1]

$$\arg\min_{u\in BV(\Omega)} \|u\|_{TV(\Omega)} + \frac{\lambda}{2}\int_{\Omega\setminus D}(f(x) - u(x))^2 dx \tag{6}$$

Using the same approach as the demo to solve the functional minimization, I obtained the following inpainted image (Figure 5a).


5c. Damaged


5a. Inpainted

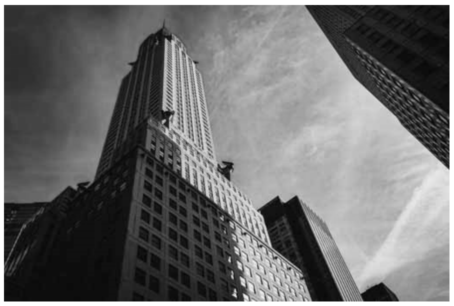

5b. Original

Figure 5. Inpainted vs. Original
(Getreuer's Method)

In this demo, I used $\lambda = 50$. However, different from Getreuer's implementation, I did not normalize the image pixel to be within $[0, 1]$ and leave it at $[0, 255]$. Therefore, the $\lambda = 50$ is equivalent to $\lambda = 50 \times 255 = 12750$ in Getreuer's paper. This λ is within the recommended range [1].

Using this method, we got a better-inpainted image, almost identical at first glance, where the edges are more defined and the image is not overly smooth. However, this method is unable to recover the image texture [1].
Getreuer(2012) also proposed an algorithm to solve this minimization problem using Split Bregman [1, page. 149-151]. However, due to the time constraints and my lack of understanding, I was not able to implement it.

4 Reflection

This project is challenging since the math related in Total Variation is not easy to wrap my head around. Because of that, I spent a lot of time on the definition of Total Variation, and how it can be used in Image Inpainting.
Moreover, because the math is complex, implementation of the algorithm is

also hard. I have spent quite some time to find the package to solve the functional minimization problem.
Once I understand the concept of Image Inpainting using Total Variation, I was able to carry out the project.
Through this project, I learned the technique to treat an image as a smooth function which enables us to do minimization/maximization to solve image processing problem. This gives more depth to my current knowledge about image processing as well as normally, I treated images as matrices of vectors. This project also demonstrates how power calculus of variation is in solving real-world problems.
Overall, the project was fun, and I think I learned a lot about image processing, specifically, image inpainting. Moreover, I learned how the image processing problem could be turned into a minimization/maximization problem that can solved using math techniques.

References

[1] Pascal Getreuer. Total Variation Inpainting using Split Bregman. *Image Processing On Line*, 2:147–157, 2012. https://doi.org/10.5201/ipol.2012.g-tvi.

[2] Leonid I. Rudin, Stanley Osher, and Emad Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D Nonlinear Phenomena*, 60(1-4):259–268, November 1992.

Appendices

A Codes And Images Used

The implementation is available on the following public GitHub repository.
https://github.com/khaidongduc/tvinpaint
Each scripts takes about 2 minutes to inpaint a 512×341 image.

B Video And Project Link

The project's folder link is available on Google Drive and is shared with Union College.
https://drive.google.com/drive/folders/14x3zFk0Ng5i9HDnVc3lEQGIPQJjkqoT4?usp=sharing
The folder contains:

- *238_project_final.pdf* The description of the assignment
- *project2_latex.zip* the zip file containing the latex project
- *project2_final.pdf* this pdf file
- *project2_vid.mp4* the video of the project

The standalone video link is
https://drive.google.com/file/d/1oHvCX3_XtiOo8o7iY2fBLAmcn27rjea5/view?usp=sharing