Watermark Removal and Inpainting

Lily Ren, Tina Shen, Xinyue Zhang

May 2024

1 Abstract

Watermarking has emerged as an effective means of asserting credit and personal ownership in the digital landscape of the twenty-first century. However, it often compromises the integrity of the underlying image, detracting from the viewer's experience. In response to this challenge, our project draws upon current literature, leveraging (i) detection and (ii) inpainting techniques to restore the desired images.

Our approach begins by accurately locating the watermark within the image. We then utilize this information to implement total variation inpainting and/or Mumford-Shah inpainting, leveraging the known indices of the watermark for precise restoration.

2 Background

The rapid development of Internet technology promotes the extensive sharing of information, where images play a vital role in this exchange. Visible watermark technology is commonly used for copyright protection of online images. However, this can be diminish the viewing experience. To address this, we introduce a mathematical framework specifically for the detection and removal of visible watermarks in images, consisting of two main components: Watermark Detection and Inpainting. To detect the watermark, we apply gradient descent to identify watermarks by analyzing patterns and inconsistencies in images. Image inpainting aims to fill in the information for the damaged area of an image, and restore the image to its near-original state. We implement image inpainting base on TV Regularization and Mumford-Shah model, respectively. Our dataset used is the 50 cat images from Kaggle public dataset.

The Split Bregman method[4], derived initially from Bregman iterations and the Rudin-Osher-Fatemi (ROF) model for Total Variation, has emerged as a robust and versatile technique for image inpainting. By incorporating principles from convex optimization and iterative algorithms, Split Bregman effectively addresses the challenges posed by nonlinear systems encountered in Total Variation inpainting. Its ability to handle complex image structures and efficiently restore missing regions has made it a mature and widely-used method in the field of image processing. Furthermore, Split Bregman offers computational advantages, enabling the inpainting process to be performed with reasonable efficiency, even on large-scale images.

On the other hand, the Mumford-Shah inpainting model is another powerful technique in inpainting technique, used to recover damaged areas in an image by approximating the image with a piecewise smooth function.

3 Methodology

3.1 Watermark Detection

We estimate the watermark image, alpha matte, and original images using numerous observed examples. Our approach capitalizes on data redundancy. Initially, we extract consistent image structures from the collections to derive a preliminary estimate of the matted watermark and identify the watermark region in all images. Subsequently, we address an optimization problem to separate the matted watermark into its image and alpha matte components.

3.1.1 Attack on watermarked image collections

Define the watermarked image J mathematically as a combination of the watermark W, the original image I:

$$J(p) = \alpha(p)W(p) + (1 - \alpha(p))I(p), \tag{3.1}$$

where $p = (x, y)$ is the pixel location, α specifies the transparency of the watermark at each pixel. Note we assume that watermarks are added in a consistent manner to many images. Then for a collection of image

For a collection of images, I_k,

$$J_k = \alpha W + (1 - \alpha)I_k, \quad k = 1, ..., K \tag{3.2}$$

Given $\{J_k\}_{k=1}^{K}$, our purpose is to recover W, α, and $\{I_k\}_{k=1}^{K}$.

3.1.2 Watermark Estimation

Calculate the median of the watermarked image gradients:

$$\nabla \widehat{W}_m(p) = \text{median}_k \left(\nabla J_k(p) \right), \text{ at each pixel location } p \tag{3.3}$$

Compute the expectation $E\left[\nabla J_k\right]$ from the equation (3.2), we have

$$\begin{aligned} E\left[\nabla J_k\right] &= E\left[\nabla W_m\right] + E\left[\nabla I_k\right] - E\left[\nabla\left(\alpha I_k\right)\right] \\ &= \nabla W_m + E\left[\nabla I_k\right] - \nabla \alpha E\left[I_k\right] - \alpha E\left[\nabla I_k\right] \\ &= \nabla W_m - \nabla \alpha E\left[I_k\right] \end{aligned}$$

It follows that $E\left[\nabla J_k\right]$ approximates the gradients of the matted watermark, W_m, except pixels shifted by $\nabla \alpha E\left[I_k\right]$. Hence, as the number of images K increases, the equation (3.3) converges to $W_m = \alpha W$. Then we crop the watermark to remove boundary regions by processing the gradients of an image to identify and isolate regions with high gradient magnitudes, which indicate potential watermark boundaries. Specifically, compute the magnitude $\nabla \widehat{W}_m$ and using a threshold to determine which gradient values are considered part of the watermark.

3.1.3 Watermark Detection

We assume that the watermark position is not fixed; instead, we define a rough bounding box around the watermark area. We then use the gradients within this bounding box as an initial estimate for the watermark gradients.

We detect the watermark by applying the Canny edge detector and chamfer matching. Given the current estimate $\nabla \widehat{W}_m$, we identify the watermark in each image using the Chamfer distance, which provides the distance from each pixel to the nearest edge. Utilizing the Canny edge detector to generate a detailed edge map and calculate its Euclidean distance transform. The watermark position is determined by the pixel with the minimum distance in the map.

3.2 Total Variation Inpaint

As we have learned in class, Total Variation (for short TV) Regularization term penalizes the magnitude of the gradient of image only on the order of 1. That is, compared to Wiener Regularization, TV regularization cares more when the variation of the image gray level is small, and cares less when the variation is big. This makes the TV regularization better in distinguishing edges.

$$\text{If} \quad R_{TV}[s] = \int_I \|\nabla s\|^{\mathbf{1}} \, dt,$$

$$\text{then} \quad \delta R_{TV}[s] = -\int_I \text{div}\left(\frac{\nabla s}{\|\nabla s\|}\right) \omega \, dt$$

where I is the image domain.

Putting the R_{TV} term back to our E[s], we will get a series of formulas that we can use to find the critical point. Equivalently, we can get to this final system of equation:

$$\textbf{Want To Solve:} \quad h_{-} * (g - h * s) - \frac{\lambda}{2}\text{div}\left(\frac{\nabla s}{\|\nabla s\|}\right) = 0$$

which is not a linear system that can be solved by doing Fourier Transform. Therefore, we need to use a different approach to find the minimal value.

3.2.1 Split Bregman

To find the gradient of the image, Split Bregman proposes the one-step forward differentiation formula (which we also covered in class). We will express the matrix $f_{\mathrm{i,j}}$ as the forward difference.
Since we are only considering inpainting, the data fidelity term will be the total variation of the image, which will be expressed as the sum of vector magnitude $\|\nabla u_{i,j}\|$ over all pixels.

$$D[u] = \sum_{i,j} \|\nabla u_{i,j}\|$$

From now on, we will use u as the updated image in each iteration.
We can thus turn our problem into the form:

$$\begin{cases} \underset{d,u}{\arg\min} & \sum_{i,j} |d_{i,j}| + \frac{1}{2}\sum_{i,j} \lambda_{i,j}(f_{i,j} - u_{i,j})^2 \\ \text{such that} & d = \nabla u \end{cases}$$

Note that the reason why λ is a function with index i and j is that we would use it as an indicator function, where we would set $\lambda = 0$ for pixels that are not identified as watermark and any positive values for any pixels within the watermark region.
Using Bregman iterations[5], we can turn the formulas into two subproblems, and each will tackle one of the variables we want to minimize (in other words, one is the d problem, and one is the u problem). We would use Bregman distance formula $D^b{}_E(u, u^k) = E(u) - \langle b^k, u - u^k\rangle$ to measure the difference between the iterates of these subproblems, allowing them to be updated independently.

$$\textbf{Initialization:} \quad u_0 = f, \quad \text{and} \quad d_{0,x} = d_{0,y} = b_{0,x} = b_{0,y} = 0$$

$$b^{k+1} = d^k + \nabla u^{k+1} - b^{k+1}$$

We will now go deeper into the u and d solutions.

3.2.2 u subproblem

To solve for the u subproblem, we will use Gauss–Seidel solution[7], which is cited below. It is essentially the same operation over u, d, and b variables.

$$G_{k,i,j} = \frac{\lambda}{\mu + 4\lambda} \left(u^{k+1}_{i+1,j} + u^k_{i-1,j} + u^{k+1}_{i,j+1} + u^k_{i,j-1} + d^k_{x,i-1,j} - d^k_{x,i,j} \right. \\ \left. + d^k_{y,i,j-1} - d^k_{y,i,j} - b^k_{x,i-1,j} + b^k_{x,i,j} - b^k_{y,i,j-1} + b^k_{y,i,j}\right) + \frac{\mu}{\mu + 4\lambda} f_{i,j} \quad (1)$$

The ultimate u will ideally be obtained either by an arbitrarily chosen tolerance level or limited number of iterations. In our case, we will stop after a fixed number of iterations.

3.2.3 d subproblem

To solve for d, shrinkage function[6] will be used. Shrinkage function performs in a manner similar to gradient descent.

$$d_{k+1,x} = \text{shrink}\left(\nabla u^{k+1}_x + b_{k,x}, \frac{1}{\lambda}\right)$$

$$d_{k+1,y} = \text{shrink}\left(\nabla u^{k+1}_y + b_{k,y}, \frac{1}{\lambda}\right)$$

3.3 Mumford–Shah Inpaint

3.3.1 Overview and Energy Function

The central concept of Mumford-Shah inpaint method involves minimizing an energy function, expressed as:

$$J[u] = \int_{\Omega \setminus D} \lambda \frac{1}{2}(u - u^0)^2 \, dx + \int_{\Omega \setminus D} \gamma |\nabla u|^2 \, dx + \alpha H^1(\Gamma)$$

Symbol	
u	represents the inpainted image.
u^0	is the original image before damage.
Ω	denotes the image domain, and D the damaged area.
λ, γ, α	are parameters balancing fidelity to data, smoothness, and edge length.
$H^1(\Gamma)$	denotes the Hausdorff measure of the discontinuity set Γ, representing edges.

Table 1: Variables and parameters in the Mumford-Shah model

where,

And this can be interpreted as follows:

Data Fidelity Term: $\int_{\Omega\backslash D} \lambda \frac{1}{2}(u - u^0)^2 \, dx$

This term ensures that the inpainted image u stays close to the original image u^0 in undamaged regions $\Omega\backslash D$. The parameter λ controls the weight of fidelity.

Smoothness Term: $\int_{\Omega\backslash D} \gamma |\nabla u|^2 \, dx$

This term regulates the smoothness of the inpainted image by penalizing sharp changes (gradients) in u. The parameter γ adjusts the degree of smoothness, with higher values leading to smoother images.

Edge Preservation Term: $\alpha H^1(\Gamma)$

This term penalizes the length of the edges in the image, encouraging shorter and fewer discontinuities in the inpainted image. The parameter α controls the strength of edge preservation.

3.3.2 Ambrosio-Tortorelli Approximation

Solving the Mumford-Shah model directly is complex due to its non-linearity and the presence of the free discontinuity set Γ. To address this, the Ambrosio-Tortorelli Approximation introduces a phase field approach, simplifying the discontinuity into a smooth function. This approximation uses an indicator function χ, close to 0 near edges and 1 elsewhere. Thus, the energy functional becomes:

$$J_{AT}[u, \chi] = \int_{\Omega} \left(\chi^2 + \epsilon\right) |\nabla u|^2 \, dx + \frac{1}{\epsilon} \int_{\Omega} (1 - \chi)^2 \, dx$$

where ϵ is a small parameter controlling the transition between 0 and 1. [1]

3.3.3 Iterative Solution

The Euler-Lagrange equations for u and χ derived from the Ambrosio-Tortorelli energy functional are used to find the stationary points by varying u and χ respectively:

$$\frac{\delta J_{AT}}{\delta u}: \quad \mathbb{1}\nabla u - \operatorname{div}((\chi^2 + \epsilon)\nabla u) = 0$$

$$\frac{\delta J_{AT}}{\delta \chi}: \quad -\alpha\chi|\nabla u|^2 + \beta(-2\epsilon\delta\chi + \frac{\chi - 1}{2\epsilon}) = 0$$

These nonlinear equations are solved iteratively using linear approximations:

$$A_{u,\chi} = 1 + \frac{2\epsilon\alpha}{\beta}|\nabla u|^2 - 4\epsilon^2\Delta$$

$$B_\chi = -\operatorname{div}((\chi^2 + h(\epsilon))\nabla V) + \frac{1}{\alpha}V$$

where V is an iteration variable, and $h(\epsilon)$ is a ϵ dependent smoothing function.

In practical terms, this approach allows for efficient image inpainting. The actual algorithm is implemented in Python, and the results will be demonstrated later in this report.

4 Results

In this project, we have implemented watermark detection and watermark removal methods.

Cropping $\nabla\widehat{W}_m$ to remove boundary regions may be demonstrated by comparing reconstructed images from original gradients and cropped gradients in Figure 1. For Watermark Detection, we mark a rough bounding box around the watermark in one of the images and print the watermark we detected with the cropped gradients in Figure 2. The start point of the detected watermark is given by $(72, 39)$, and the end point is $(21 + 72, 128 + 39) = (93, 167)$.

Carrying out TV regularization inpaint, we can see a fairly good removal of the

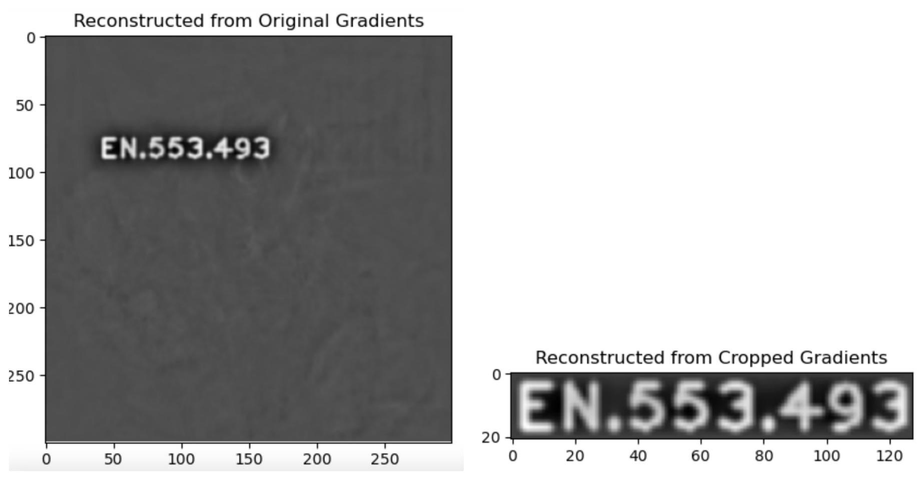

Figure 1: (i) Reconstruct the watermark with the original gradients (ii) Reconstruct the watermark with the cropped gradients

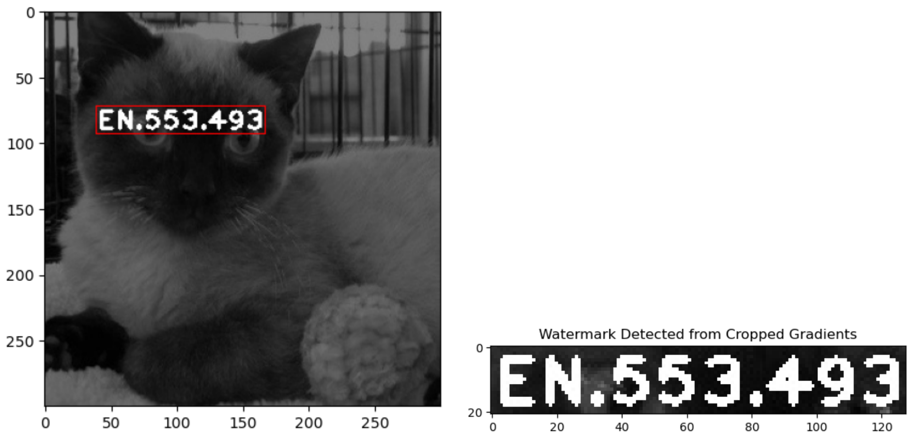

Figure 2: (i) A rough bounding box around the watermark in a single image. (ii) Watermark detected from cropped gradients.

masked (that is, detected) region, and it recovered the image in its matching gray level.

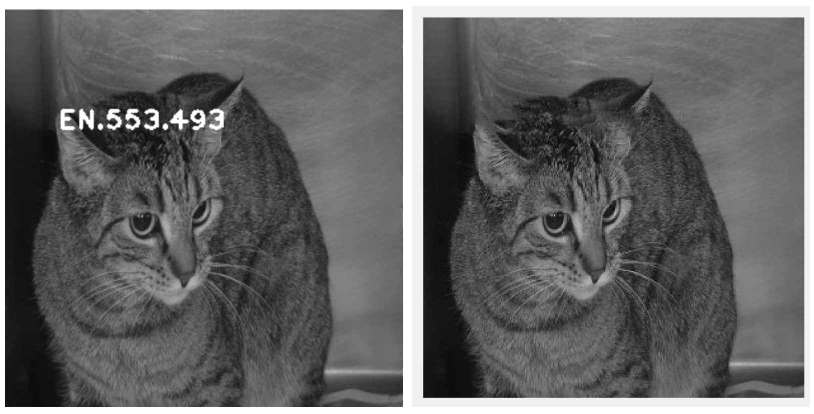

Figure 3: Before and after the application of TV Split Bregman inpainting

In the implementation of the Mumford-Shah inpainting model using the revised gradient computation functions, we can see improvements in the quality of image restoration, especially around the edges of the inpainting area. The algorithm effectively minimized the Mumford-Shah energy functional, resulting in smoother transitions within the inpainted areas while preserving the integrity of the natural edges and fine details of the original image. The results from the rectangular watermark removal are illustrated below:

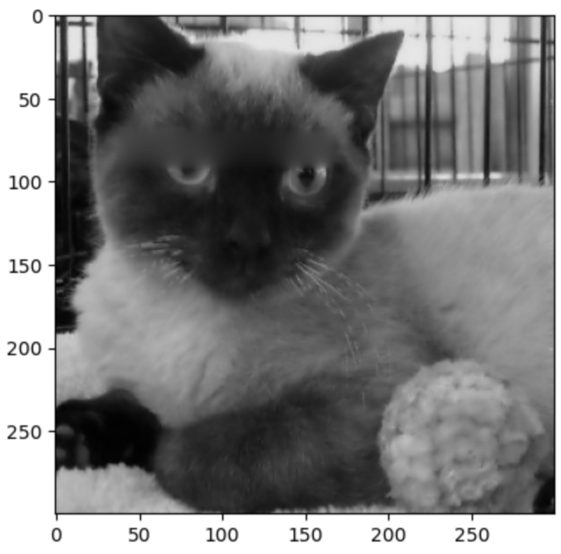

Figure 4: Inpainting with Mumford-Shah

Additionally, targeting only the textual watermark areas for inpainting gives better results. This approach, however, needs an improved watermark detection method to precisely find the watermark text regions. The enhanced outcomes with targeted inpainting can be shown in the following figure:

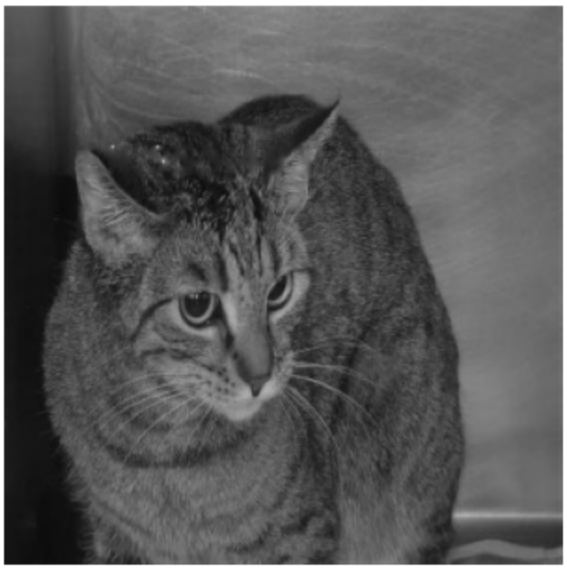

Figure 5: Inpainting with Mumford-Shah and Textual Watermark

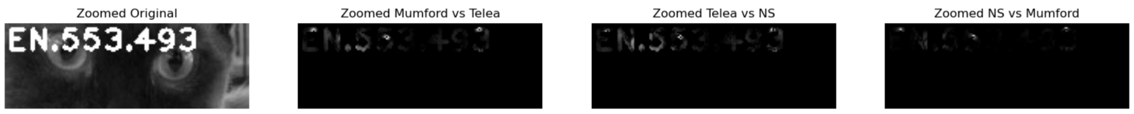

Figure 6: Absolute Diffrences between Mumford-Shah and Python Built-in Inpainting Method, Zoomed

5 Future Directions

Looking ahead, we will focus on several key areas to improve our image inpainting and watermark removal methods.

1. Experiment of Regularization Parameters:

We aim to experiment with a wider range of regularization parameters. This will allow us to finetune our models to handle more complex scenarios, such as varying watermark intensities and locations. By systematically varying parameters like the smoothness constant and the fidelity term, we can develop a more nuanced approach that adapts to the characteristics of the watermark and the image.

2. Enhancement of Quantitative Evaluation Metrics:

Our current methodology primarily uses visual assessment. Moving forward, we plan to incorporate additional numerical metrics such as the Structural Similarity Index (SSIM) and the Peak Signal to Noise Ratio (PSNR). These metrics will provide an additional perspective on our results, combining mathematical accuracy with visual quality assessments. Such an approach will ensure that our watermark removal techniques not only reduce error in a statistical sense but also enhance the subjective visual experience of the end users.

3. Comprehensive Testing with Varied Watermarks:

We propose to create a detailed database of watermarks with a variety of characteristics, including size, shape, transparency, and positional variability within the image. This database will serve as a benchmark for rigorously testing our algorithms under other conditions.We can better test our approaches and refine the models to be more adaptable.

6 Conclusion

In conclusion, our experiments with watermark detection and removal have demonstrated fairly ideal results, particularly with the integration of the Mumford-Shah model and TV Regularization. By enhancing our evaluation metrics and broadening the range of test conditions, we aim to develop a more comprehensive detection and inpaint model that showcases the our understanding of digital image processing.

7 Code

Github link: https://github.com/aenorhabditis6/mathematical-image-analysis/tree/main

References

[1] Carola-Bibiane Schönlieb. *Partial differential equation methods for image inpainting.* Cambridge University Press, 2015.

[2] T. Dekel, M. Rubinstein, C. Liu, and W. T. Freeman. On the effectiveness of visible watermarks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2146-2154, 2017.

[3] T. Kalker, J. P. Linnartz, and M. van Dijk. Watermark estimation through detector analysis. In *Proceedings 1998 International Conference on Image Processing. ICIP98*, Vol. 1, pp. 425-429, IEEE, October 1998.

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

[5] Bregman, L.M. (1967). The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR Computational Mathematics and Mathematical Physics*, 7(3), 200-217. ISSN 0041-5553. https://doi.org/10.1016/0041-5553(67)90040-7.

[6] T. Goldstein and S. Osher. (2009). The Split Bregman Method for L1-Regularized Problems. *SIAM J. Imaging Sciences.* 2. 323-343. 10.1137/080725891.

[7] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, Cambridge, UK, 2004.

Thank you for a great semester!