Implementation of the Polyakov Action Minimization for Efficient Color Image Processing

Mercè Nachón Moreno

Abstract

The Laplace-Beltrami operator is an extension of the Laplacian. It was proven to be useful for color image processing as it models a meaningful coupling between the color channels. This is expressed in the Beltrami framework in which a color image is regarded as a map between two dimentional cartetian space to a five-dimensional, spatialchromatic (x, y, R, G, B) space.

The Beltrami filter defined by this framework minimizes the Polyakov action, adopted from high-energy physics, which measures the area of the image manifold. Where in the paper we achieve to implement'The Polyakov Action Minimization for Efficient Color Image Processing', they propose to use an augmented Lagrangian approach to design an efficient and accurate regularization framework for color image processing by minimizing the Polyakov action, extending the augmented Lagrangian framework for total variation (TV) image denoising to the more general Polyakov action case for color images, and apply the proposed framework to denoise and deblur color images.

1 Introduction

Variational, nonlinear diffusion filters have been extensively used for various image processing tasks. Numerical schemes that implement image regularization techniques are carefully designed to prioritize accuracy, stability, and computational efficiency. While many studies focus on regularization for grayscale images, in resent years the appearance of a few number of works have been made to regularize vector-valued signals.

These works describe regularization functionals which operate on vector-valued images. The Beltrami framework describes a regularizing functional, well suited for color image processing, which can be justified by the Lambertian model of color image formation. The framework considers the image as a 2-manifold embedded in a hybrid spatial-feature space. Regularization of the image in this framework is expressed as minimization of area surface. The Beltrami filter is strongly related to the bilateral filter, as well as to the non-local means filter. Minimization of the associated functional is usually done by evolving the image according to its Euler-Lagrange equation, but is computationaly constly. Another way to do it is by performing a fixed-point iteration from the Euler-Lagrange equation. Recently, several approaches were suggested for improving the speed of computation of minimizers for the Polyakov action. The latter can be done by including a Beltrami filter kernel, as well as employing vector extrapolation techniques, or operator splitting methods.

In the paper we based our implementation on, they propose to minimize the discretized Polyakov action. So in this work we will start introducing our theorical framework, which includes the explanation of the Beltrami Framework for color image regularization, expanding to the application of the coupled constrained optimization approach to regularize color images using the Polyakov action. Then we expose the issues we encountered during the implementation, followed by the solutions and the results.

2 Theoretical Framework

2.1 Beltrami Framework

Let us start by introducing the Beltrami framework for non-linear diffusion. In this framework, we are working with images as maps between two Riemannian manifolds. These Riemannian manifolds describe themselves as spaces that generalize notions of curves and surfaces in Riemannian Geometry, which also locally resemble vector spaces and are equipped with a positive definite inner product.

We denote such maps by $X : \Sigma \to M$, being Σ a two-dimensional manifold parameterized by global coordinates (σ_1, σ_2), and M the manifold involving the spatial features, which lies in $\mathbb{R}^{d+2}$, where d is the number of image channels. For example, a gray-scaled image can be represented as a surface embedded in $\mathbb{R}^3$. In this case the map X is

$$X(\sigma_1, \sigma_2) = (\sigma_1, \sigma_2, I(\sigma_1, \sigma_2))$$

where I is the image intensity. In this example, we just had one image intensity, but usually will have an image intensity per channel, ie., given an image of d intensities we will have the set $\{I_i\}_{i\in\{1,\dots,d\}}$ of intensities.

By introducing the metric $g_{i,j}$ over M, we can find the inverse metric elements denoted by $g^{i,j}$, and the notation for the determinant will be $\mathbf{g} = det(g_{i,j})$. Designate by (Σ, g) the image metric space, and by (M, h) the space-feature space. The functional $S[X]$ characterizes the mapping $X : \Sigma \to M$, and is defined to be

$$S[X, g_{i,j}, h_{a,b}] = \int d^m\sigma\sqrt{g}\,||dX||^2_{g,h}\,, \tag{1}$$

where m is the dimension of Σ, g is the determinant of the image metric, and the range of indices is $i, j \in \{1, \dots, dim(\Sigma)\}$ and $a, b \in \{1, \dots, dim(M)\}$. The operator $||dX||^2_{g,h}$ is defined as

$$||dX||^2_{g,h} = (\partial_{\sigma_i} I_a)\, g^{i,j}\, (\partial_{\sigma_i} I_a)\, g^{i,j}$$

So, given that we are working in image processing, and the usual choice of coordinates is Cartesian, we will use (x, y) instead of (σ_1, σ_2). Since this means that $dim(\Sigma) = 2$, by assuming $h_{a,b} = \delta_{a,b}$ we can call the functional (1) the Polyakov action, as known in string theory. In the case of color images, having also the spatial space assumed to be Cartesian, the metric becomes

$$g_{i,j} = \begin{pmatrix} 1 + \beta^2 \sum_{a=1}^3 (\partial_x I_a)^2 & \beta^2 \sum_{a=1}^3 \partial_x I_a \partial_y I_a \\ \beta^2 \sum_{a=1}^3 \partial_x I_a \partial_y I_a & 1 + \beta^2 \sum_{a=1}^3 (\partial_x I_a)^2 \end{pmatrix},$$

where $\beta > 0$ is the parameter that determines the ratio between coordinate spaces. It follows $S(X)$ becoming

$$S(X) = \int \sqrt{g} dx dy$$

with

$$g = 1 + \beta^2 \sum_{a=1}^3 ||\nabla I_a||^2 + \frac{\beta^4}{2} \sum_{a,b=1}^3 ||\nabla I_a \times \nabla I_b||^2\,.$$

Here the functional S can be minimized for $\{I_a\}_{a\in\{1,2,3\}}$ by the Euler-Lagrange equations giving us

$$-\frac{1}{2\sqrt{g}} h^{ab} \frac{\delta S}{\delta I_b} = \frac{1}{\sqrt{g}} \mathrm{div}\big(\sqrt{g} g^{i,j} \nabla I_a\big). \tag{2}$$

This equation can be re-written as $\delta_t I_a = \nabla_g I_a$, where ∇_g generalizes the Laplacian to manifolds, can be called the Laplace-Beltrami operator.

Since we will be working with color color image restoration, in makes sense to generalize such functional as the family of functionals

$$\int \sqrt{\beta_1 + \beta_2 \sum_{a=1}^3 ||\nabla I_a||^2 + \beta_3 \sum_{a,b=1}^3 ||\nabla I_a \times \nabla I_b||^2}. \tag{3}$$

In the variational framework, the reconstructed image minimizes a cost functional of the form

$$\Phi = \frac{\alpha}{2} \sum_{a,b=1}^{3} ||KI_a - I_a||^2 + S(X),$$

where K is a bounded linear operator that depends on what type of restoration we are making: deblurring or denoising, meanwhile α is the parameter that controls smoothness of the solution. In this project the goal is to implement the approach of optimization of the functional Φ using an augmented Lagragian method.

2.2 Augmented Lagragian Method.

Let us introduce a bit what are the augmented Lagrangian methods. They are a certain class of algorithms for solving constrained optimization problems, like the one we got. They have similarities to penalty methods in that they replace a constrained optimization problem by a series of unconstrained problems and add a penalty term to the objective, the difference is that the augmented Lagrangian method adds yet another term, designed to mimic a Lagrange multiplier.

It is usual to work with this methods. One of the best total variation regularization methods is the one which is obtained by decoupling the optimization problem

$$\min_u \int |\nabla u| + \frac{\alpha}{2} ||Ku - f||^2, \tag{4}$$

where f is our modified image, into a constrained optimization problem

$$\min_{u,\mathbf{Q}} \int |\mathbf{Q}| + \frac{\alpha}{2} ||Ku - f||^2, \quad \mathbf{Q} = \nabla u, \tag{5}$$

being $\mathbf{Q}$ an auxiliary field parallel to the gradient of u. This constraint is then incorporated using an augmented Lagrangian penalty function of the form

$$\rho_{\mu,r}(u, \mathbf{Q}) = \mu^T (\nabla u - \mathbf{Q}) + \frac{r}{2} ||\nabla u - \mathbf{Q}||^2,$$

penalty that enforces the restriction $\mathbf{Q} = \nabla u$.

In this case, by working with color images in the Polyakov action, we replace the gradient norm penalty used in total variation regularization for the functional exposed in (1). In this way we obtain, through equation (5), that

$$\int \sqrt{1 + \beta^2 \sum_{i\in\{R,G,B\}} ||\mathbf{Q}_i||^2 + \frac{\beta^4}{2} \sum_{i\in\{R,G,B\}} \sum_{j\neq i} ||\mathbf{Q}_i \times \mathbf{Q}_j||^2}, \tag{6}$$

being β the spatial-intensity aspect ratio and $\{\mathbf{Q}_i\}_{i\in\{R,G,B\}}$ are the components of the field $\mathbf{Q}$, parallels to the gradient of each image channel red, green and blue (R, G, B), respectively. We then extend the rest of the functional to the vectorial (per-pixel) case, obtaining

$$\mathcal{L}_{BEL}(u, \mathbf{Q}, \mu) = \int \sqrt{\begin{array}{c} 1+\beta^2 \sum_{i\in\{R,G,B\}} ||\mathbf{Q}_i||^2 + \frac{\beta^4}{2} \sum_{i\in\{R,G,B\}} \sum_{j\neq i} ||\mathbf{Q}_i \times \mathbf{Q}_j||^2 + \\ \sum_{i\in\{R,G,B\}} \mu_i^T (\mathbf{Q}_i - \nabla u_i) + \frac{r}{2} \sum_{i\in\{R,G,B\}} ||\mathbf{Q}_i - \nabla u_i||^2 \end{array}},$$

which corresponds to Beltrami regularization. Here we have that u and μ are replace by their channel equivalents, $\{u_i\}$ and $\{\mu_i\}$, for $i \in \{R, G, B\}$. So, solving the minimization problem by discomposing it into subproblems, we arrive to the iterative algorithm in which in every iteration k we solve for u and μ like follows:

$$u_i^k = \mathcal{F}^{-1} \left(\frac{\alpha \mathcal{F}(K^*)\mathcal{F}(f_i) - \mathcal{F}(D_x^-) \left((\mu_i^1)^k + r(p_i)^k \right) - \mathcal{F}(D_y^-) \left((\mu_i^2)^k + r(q_i)^k \right)}{\alpha \mathcal{F}(K^*)\mathcal{F}(K) - r\mathcal{F}(\Delta)} \right)$$

and

$$\mu_i^k = \mu_i^{k-1} + r \left(\mathbf{Q}_i^k - (\nabla u_i)^k \right)$$

where $\mu^k = \left(\left(\mu_i^1\right)^k, \left(\mu_i^2\right)^k\right)$, $\mathbf{Q}^k = \left(\left(p_i\right)^k, \left(q_i\right)^k\right)$,

$\mathcal{F},\ \mathcal{F}^{-1}$: represets the Fourier transform and its inververse,
$D_y^-,\ D_x^-,\ \nabla$: are the kernels of the backward derivatives and the laplatian operator, and
K^* : is the L_2 adjoint of K.

For the updating of $\mathbf{Q}$ we will use a short inner-loop of a fixed-point solver with iterative reweighted least squares (IRLS). The goal is to optimize the following function for for $\mathbf{Q}_i$

$$\sqrt{1+\beta^2\sum_i\left(p_i^2+q_i^2\right)+\frac{\beta^4}{2}\sum_i\sum_{j\neq i}\left(p_iq_j-q_ip_j\right)^2}+\frac{r}{2}\sum_i||\mathbf{q}_i-\left(\nabla u\right)_i||^2+\sum_i\left(\mu_i^k\right)^T\left(\mathbf{q}_i-\left(\nabla u\right)_i\right),$$

where $\left(\nabla u\right)_i = \left(\partial_x u_i, \partial u_i\right)^T$ denotes the gradient of each channel. This is done by updating p_i as

$$\frac{2\left(\frac{1}{\beta^2}+\frac{1}{2}\sum_{j\neq i}(q_j^l)\right)p_i-\frac{1}{2}\left(\sum_{j\neq i}(q_j^l)(p_j^l)(q_i^l)\right)}{\sqrt{\frac{1}{\beta^4}+\frac{1}{\beta^2}\sum\left((p_i^{l-1})^2+(q_i^{l-1})^2\right)+\sum_a\sum_{j\neq i}\left((p_i^{l-1})(q_j^{l-1})+(q_i^{l-1})(p_j^{l-1})\right)^2}}+r\left((p_i^l)-\partial_x u_i)+(\mu_i^k)_x\right)=0.$$

and similarly for q_i, where l denotes the IRLS iteration number.

3 Implementation

During the implementation of the Algorithm 1 called *Augmented Lagrangian optimization of the Beltrami framework* on [1], we faced some problems trying to optimize $\mathbf{Q}$, given that is baggily described how to do it, and referred to a more detailed explanation in an, now deleted, technical report. Also, ones we understood a bit more what they were asking for, more usually that not, the updating gave me the error '*true_divide*' in python, resulting in black images.

Giving that, we were given the option to update $\mathbf{Q}$ per pixel like so:

$$\mathbf{Q}_{i,j}=\begin{cases}\left(1-\frac{1}{r}\frac{1}{|\mathbf{w}_{i,j}|}\right)\mathbf{w}_{i,j}, & |\mathbf{w}_{i,j}|>\frac{1}{r},\\ 0, & |\mathbf{w}_{i,j}|\leq\frac{1}{r},\end{cases}$$

where

$$\mathbf{w}=\nabla u+\frac{\mu^k}{r}.$$

This formula is taken from the *Augmented Lagrangian method for the ROF* algorithm stated in [2].

3.1 Results

Following, we will show the results obtained for gray-scaled and color images. In the latter, we added a stopping condition stating that if

$$\frac{||u^k-u^{k-1}||}{||u^{k-1}||}<\varepsilon$$

where u^{k-1} is lasts iteration image, then we return u^k. Saying this, the number of iterations made for every image varies. But in both cases we stick with 0.05 as our choice of r. Meanwhile we tried using different α values, to see how the outcome varies. In both cases we used the same three color images but changed the color scale if the case warranted it. Also, during denoising task we added noise to the images with the Gaussian noise of $\sigma = 20$ and K was the identity kernel. For the debluring task we used the Gaussian blur to make the images blurry and as K in the algorithm.

Figure 1: Used images, labeled from left to right butterfly, lion, astronaut.

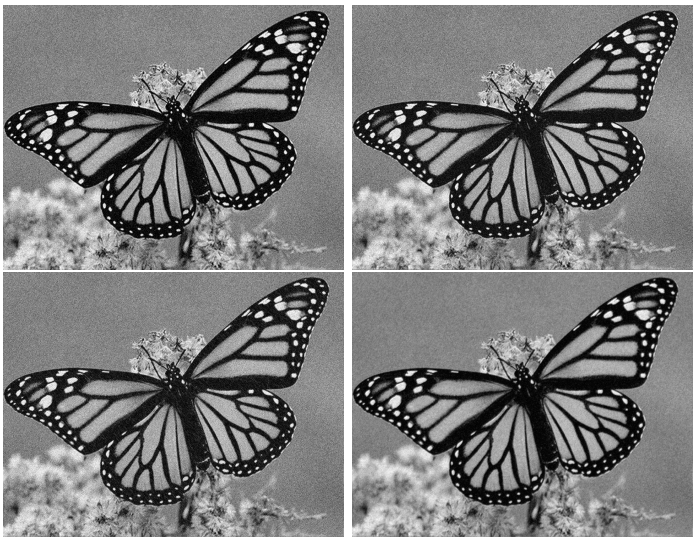

Figure 2: Fist image is the noisy one, then after AL with $\alpha = 1$, $\alpha = 0.15$ and $\alpha = 0.05$

The images used are the ones exhibit in Figure (1), and their variations in gray-scale.

For the gray-scale denoising model we obtained the following results for the butterfly image. We obtained the times and PSNR values for each α, as exposed in Table (1).

Now, for color images. This was the bulk of we were expected to do, so for them we will expose and compere the results given for each of the three images (3, 4, 5) with the same α's.

We now show the times and PSNR for each image in Table (2).

For the debluring task we did not get satisfing results, but either way you can see the output of the algorithm for the butterfly image (6) given the parameters $\alpha = 8$ and $r = 10$, in this occasion we obtain better results by updating r in each iteration by multipliying it by a variable $\gamma > 1$, as in noted in [1].

α	time	PSNR
1	13.8184	16.03
0.15	13.5589	15.57
0.05	13.5562	22.22

Table 1: Comparison of α's.

Figure 3: Results obtain for the butterfly image for each different α value.

Figure 4: Results obtain for the lion image for each different α value.

Image	Average time	Best PSNR
butterfly	1.7928	14.48
lion	18.1992	19.44
astronaut	3.5422	15.47

Table 2: Comparison of color images.

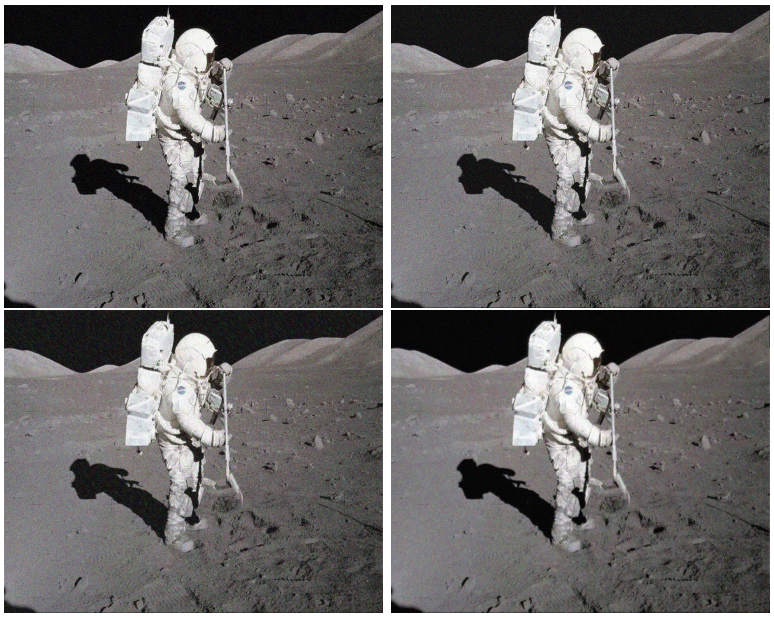

Figure 5: Results obtain for the astronaut image for each different α value.

Figure 6: Results obtain for the butterfly image for debluring.

4 Conclusions

We tried to implement an extension of the augmented Lagrangian method for color image processing with Beltrami regularization implementing a slight change. Unlike existing techniques, the method we wanted to implement discretizes the functional itself, rather than the resulting optimality conditions or minimizing flow. We presented some visual results and charts compering times and $PSNR$. In our results visually we could agree that the last image in every set, for the denoising task, appears to have the better result. Even though we did not applied the paper as it was intended, the result seem to be good and satisfying.

References

[1] Rosman, G., Tai, X.-C., Kimmel, R., Dascal, L.: Polyakov action minimization for efficient color image processing, Technion (2010)

[2] Tai, X.-C., Wu, C.: Augmented Lagrangian method, dual methods and split Bregman iteration for ROF model. In: SSVM, pp. 502–513 (2009)

[3] Tai, X.-C., Wu, C.: Augmented Lagrangian Method, Dual Methods, and Split Bregman Iteration for ROF, Vectorial TV, and High Order Models. In: SSVM, pp. 300–339 (2010)

[4] Kimmel, R., Malladi, R., Sochen, N.: Images as embedding maps and minimal surfaces: Movies, color, texture, and volumetric medical images. Int. J. of Comp. Vision 39(2), 111–129 (2000)

[5] Wu, C., Zhang, J., Tai, X.-C.: Augmented Lagrangian method for total variation restoration with non-quadratic fidelity. CAM Report 09-82, UCLA (December 2009)

[6] Vogel, C. R., Oman, M. E.: Iterative methods for total variation denoising. In: SSVM, pp. 227-238, (January 1996)

[7] Lü, T., Neittaanmaäki, P., Tai, X.-C.: A parallel splitting up method and its application to Navier-Stokes equations. Applied Mathematics Letters 4(2), 25–29 (1991)

[8] Wikipedia: Embedding. Retrieved from en.wikipedia.org, (2023)

[9] Wikipedia: Riemannian manifold. Retrieved from en.wikipedia.org, (2023)

[10] Wikipedia: Variedad. Retrieved from es.wikipedia.org, (2023)

[11] Wikipedia: Augmented Lagrangian method. Retrieved from en.wikipedia.org, (2023)