Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

MATHEMATICAL FOUNDATIONS OF COMPUTER GRAPHICS AND VISION

EXERCISE 5 - RIGID TRANSFORM BLENDING AND VARIATIONAL METHODS

Handout date: 09.05.2016
Submission deadline: 29.05.2016, 23:59
Demo date: 30.05.2016, at the exercise session

General Rules

Plagiarism note. Copying code (either from other students or from external sources) is strictly prohibited! We will be using automatic anti-plagiarism tools, and any violation of this rule will lead to expulsion from the class.

Late submissions will not be accepted, except in case of serious illness or emergency. In that case please notify the assistants and provide a relevant medical certificate.

Software. All exercises for this course are to be implemented in the MATLAB programming language. The MATLAB distribution is available from IDES for ETH students. See the exercise session slides for hints or specific functions that could be useful for your implementation.

What to hand in. Email a .zip file of your solution to ian.cherabier@infk.ethz.ch . The file must be called "MATHFOUND16-*-firstname-familyname.zip" (replace * with the assignment number) and the subject of the email must be "MATHFOUND16". The .zip file MUST contain a single folder called "MATHFOUND16-*-firstname-familyname" with the following data inside:

- A folder named "code" containing your MATLAB code
- A README file (in pdf format) containing a description of what you've implemented and instructions for running it, as well as explanations/comments on your results.
- Screenshots of all your results with associated descriptions in the README file.

Grading. This homework is 11.7% of your final grade. Your submission will be graded according to the quality of the images produced by your program, and the conformance of your program to the expected behaviour of the assignment. The submitted code must produce exactly the same images included in your submission. Code that does not run will receive a null score.

To ensure fairness of your grade, you will be asked to briefly present your work to the teaching assistants. Each student will have 3-4 minutes to demo their submission and explain in some detail what has been implemented, report potential problems and how they tried to go about solving them, and point the assistants to the code locations where the various key points of the assignments have been implemented. See above for the scheduled demo date for this particular assignment.

Goal of this exercise

In this homework, you will apply variational methods to solve two fundamental problems in computer vision, segmentation and inpainting. You will specifically use the primal dual algorithm, which is described in [1]. We remind you the fundamentals of the primal dual algorithm in this section.

The primal dual algorithm is used to solve saddle point problems of the form:

$$\min_{x\in X}\max_{y\in Y} \langle Kx, y\rangle + G(x) - F^*(y) \tag{1}$$

where $K : X \to Y$ is a linear operator, $G : X \to \mathbb{R}$ and $F* : Y \to \mathbb{R}$ two convex, lower semi-continuous, proper functions. F^* is the convex conjugate of F [1]. Problem (1) is the primal-dual form of the following energy minimization problem:

$$\min_{x\in X} F(Kx) + G(x) \tag{2}$$

Therefore it is often a good idea when facing minimization problems such as (2) to introduce dual variables and formulate the problem in the form of (1). It allows to use the primal dual algorithm, shown in algorithm 1.

Algorithm 1 Primal Dual algorithm [1]

- **Initialization:** Choose $\sigma, \tau > 0$, $\theta \in [0, 1]$, $(x^0, y^0) \in X \times Y$, and set $\bar{x}^0 = x^0$
- **Iterations:** $(n > 0)$, update x^n, y^n and $\bar{x}^n$ as follows

$$\begin{cases} y^{n+1} &= \text{prox}_{\sigma F^*}\left(y^n + \sigma K\bar{x}^n\right) \\ x^{n+1} &= \text{prox}_{\tau G}\left(x^n - \tau K^* y^{n+1}\right) \\ \bar{x}^{n+1} &= x^{n+1} + \theta\left(x^{n+1} - x^n\right) \end{cases}$$

where K^* is the adjoint operator of K, and the prox operator is introduced, defined as:

$$\text{prox}_{\tau G}(x) = \arg\min_y \left\{ \frac{1}{2} \parallel y - x \parallel^2 + \tau G(y) \right\} \tag{3}$$

In [1], the authors prove the convergence of the algorithm for $\theta = 1$, under hypothesis for the parameters.

Total variation and primal dual algorithm. In this homework you will explore applications of the primal dual algorithm for solving total variation problems. Total variation optimization refers to the class of problems which use the L_1-norm of the gradient of the unknown as a regularizer[2].

[1] The convex conjugate of F is defined as $F^* : Y \to \mathbb{R}$, s.t. $F^*(y^*) = \sup_{y\in Y}\{\langle y^*, y\rangle + -F(y)\}$.

[2] This is called the total variation of the unknown

Given some data cost functional G, we want to find a function x that minimizes G, while having a minimal total variation:

$$\min_{x \in X} \lambda G(x) + \| \nabla x \|_1 \tag{4}$$

where λ is a regularization parameter that controls the importance of the data cost over the regularization. It can be shown that $\| \nabla x \|_1 = \max_{y \in Y} \langle \nabla x, y \rangle$ where $Y = \{ y \in D_X | \| y \|_\infty \leq 1 \}$. We denote D_X the space where ∇x lives. Thus (4) can be rewritten:

$$\min_{x \in X} \max_{y \in D_X} \langle \nabla x, y \rangle + \lambda G(x) - \delta_Y(y) \tag{5}$$

where the δ function is defined as:

$$\delta_Y(y) = \begin{cases} 0 & \text{if } y \in Y \\ \infty & \text{if } y \notin Y \end{cases} \tag{6}$$

It is used to enforce the constraint that $y \in Y$. We see that (5) is written in the form of (1), where $K = \nabla$, and $F^* = \delta_Y$. Note that the adjoint of the gradient operator is the negative divergence $\nabla^* = -\text{div}$.

Under the form (5), the total variation minimization problem can be solved using the primal dual algorithm.

Weighted TV. When doing image processing we can change the equation (4) to consider the weighted version of the total variation problem. If we have image I_0 as an input:

$$\min_{x \in X} \lambda G(x) + \frac{1}{\| \nabla I_0 \|_1 + 1} \| \nabla x \|_1 \tag{7}$$

and hence, the primal dual equation:

$$\min_{x \in X} \max_{y \in D_X} \frac{1}{\| \nabla I_0 \|_1 + 1} \langle \nabla x, y \rangle + \lambda G(x) - \delta_Y(y) \tag{8}$$

1. PART 1: Convexity of the Ruding Osher Fatemi functional

One of the key concept of the primal dual algorithm is that of convexity. In this first part you will work with this notion.

Task 1. Which of the following functions are convex? Please give a proof for your answer. If the function is non convex, please give a counter example.

- $x \mapsto \sin(x)$
- $x \mapsto x^2$
- $x \mapsto \sin(x) + x^2$

Task 2. Prove the convexity of the Rudin Osher Fatemi functional, which is used to denoise images:

$$E_{ROF}(I_u) = \int_\Omega \left[|\nabla I_u(\mathbf{x})| + \| I_u(\mathbf{x}) - I_0(\mathbf{x}) \|_2^2 \right] d\mathbf{x} \tag{9}$$

Hint:

- You can use the triangular inequality, and the fact that the sum of two convex functions is convex.

Written output for part 1.

- Answers to the questions with proofs or counter examples.

2. PART 2: Segmentation revisited

In the first part of the homework you will reuse the framework of the interactive graphcut segmentation that you used for homework 4. This time you will implement TV segmentation using the primal dual algorithm.

In order to segment an image, you will need to find a function $x : \mathcal{D}_I \to [0, 1]$ where $\mathcal{D}_I$ is the domain of image I. The function should be such that it is close to 0 in the background and close to 1 in the foreground. To do so, you will draw scribbles that will allow you to build a cost function that enforces such a behaviour.

This cost function will have the following form:

$$G(x) = \langle x, f \rangle + \delta_{[0,1]}(x) \tag{10}$$

where $\delta_{[0,1]}$ is defined as (6), and f is such that it is negative for foreground and positive for background. You can see how it makes x be 0 for background and 1 for foreground: if you want to minimize G without regularizing, you will want to choose $x_i = 0$ for positive values of f_i and $x_i = 1$ for negative values of f_i. To build f, you can use the color histograms that you created for homework 4, we will call them H_{fg} and H_{bg}. Then you will build f as:

$$f_i = \log H_{bg}(I_i) - \log H_{fg}(I_i) \tag{11}$$

for all $i \in \mathcal{D}_I$. We can then adapt (5) to the segmentation problem:

$$\min_{x \in X} \max_{y \in D_X} \langle \nabla x, y \rangle + \lambda \langle x, f \rangle + \delta_{[0,1]}(x) - \delta_Y(y) \tag{12}$$

You can now use the primal algorithm to solve (12).

Steps of this task

- Implement a code that compute the cost function using the scribbles and the color histogram.

FIGURE 1. Segmented image

FIGURE 2. Function x catpuring the segmentation

  - If you weren't able to implement a color histogram function for homework 4, we provide you a cost cost function that will only work for the given scribbles, so that you can still implement the primal dual algorithm, and verify if it works correctly.
- Implement the primal dual algorithm using the following parameters: $\tau = 0.35$, $\sigma = 0.35$, $\lambda = 0.0075$. Your results should look like figures 1 and 2.

Written output for part 2.

- Provide the segmentation of the given image
- Please write a few lines to compare primal dual TV segmentation with graph cut.

Hints.

- For $F^*(y) = \delta_Y(y)$, we have that $\text{prox}_{\sigma F^*}(u) = u'$ such that
$$u'_i = \frac{u_i}{\max(1, \parallel u_i \parallel_2)}$$
where i refers to the pixel position in the image.
- For $G(x) = \langle x, f\rangle + \delta_{[0,1]}(x)$ we have that:
$$\text{prox}_{\tau G}(u) = \min(1, \max(0, u + \tau f))$$
- When you are doing computations on images in matlab such as computing the gradient, it is better to convert them into double, using the command $double(I)$, where I is the image. After you did the computation, if you want to print the images, you need to convert them into the uint8 format, using the command $uint8(I)$.

3. PART 3: Applications of inpainting

In this section, you will apply the primal dual algorithm to inpainting, which consists in recovering missing parts of images. Inpainting has many applications. You will focus on two of them:

- Removing data from an image
- Recovering a compressed image

We begin by describing the formulation of inpainting as a total variation problem and its primal dual formulation.

3.1. **TV inpainting.**

The input data for inpainting is an image I with missing pixels. We define the inpainting region as the set of all the missing pixels, which we denote $\mathcal{I}$. The cost funtion can be naturally chosen as:

$$G(x) = \frac{1}{2} \sum_{i,j \in \mathcal{D}_I \setminus \mathcal{I}} \frac{1}{2} (I_{i,j} - x_{i,j}) \tag{13}$$

The total variation can be used as a regularizer. It allows to recover an image with preserved edges. Then (5) can be rewritten as:

$$\min_{x \in X} \max_{y \in D_X} \langle \nabla x, y\rangle + \frac{\lambda}{2} \sum_{i,j \in \mathcal{D}_I \setminus \mathcal{I}} \frac{1}{2} (I_{i,j} - x_{i,j}) - \delta_Y(y) \tag{14}$$

With this new cost function, the prox operator is now $\text{prox}_{\tau G}(u) = u'$ such that

$$u'_{ij} = \begin{cases} u_{ij} & \text{if } (i,j) \in \mathcal{I} \\ \frac{u_{ij} + \tau\lambda I_{ij}}{1 + \tau\lambda} & \text{else} \end{cases}$$

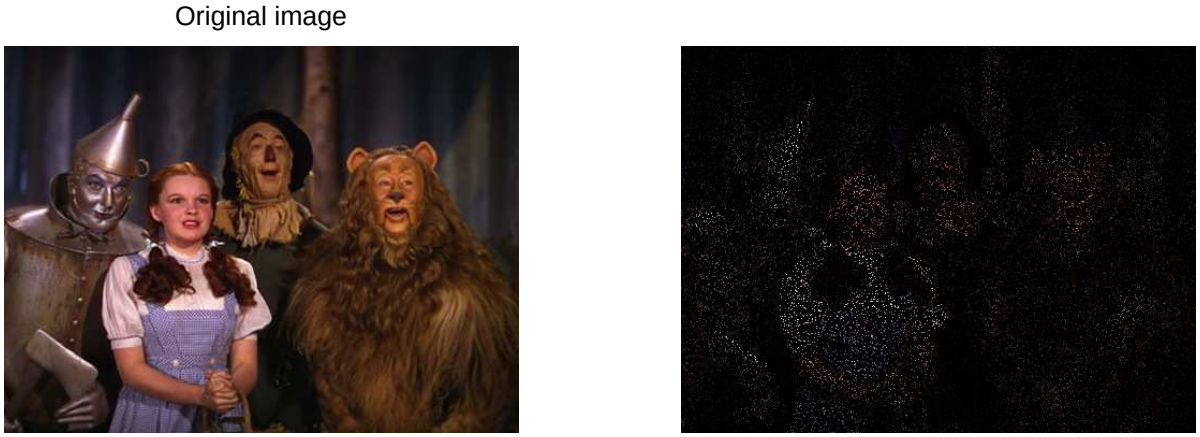

FIGURE 3. Image that you will recover with inpaintng

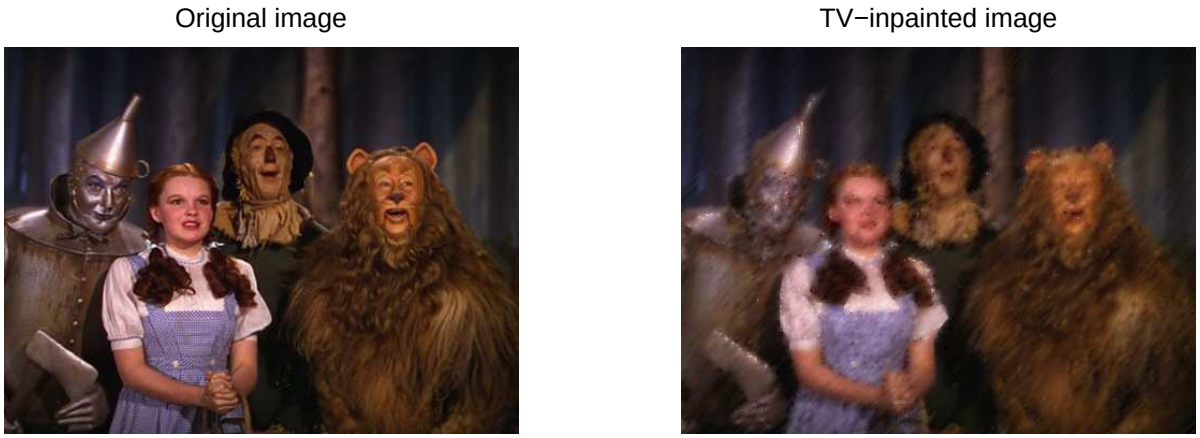

FIGURE 4. Inpainted image compared to the original image

3.2. **Task1: Recovering an image with a high number of missing pixels.** For this task, we provide you with an image from which approximately 90% of the pixels were removed. The missing pixels are represented by a mask with entries 0 and 1 which is applied to the image. Your task will be to recover the image, using the inpainting method of (14). See figure 3.

Steps of this task.

- Implement the primal dual algorithm for inpainting with the following paramers: $\tau = 1.5$, $\sigma = 1.5$, $\lambda = 5$. Results should look like figure 4.

3.3. **Task 2: Removing an object from an image.** When you use a software like photoshop to remove objects from images, you are actually using an inpaiting method. In this task, you will

FIGURE 5. Abraham Lincoln with an artifact

FIGURE 6. Abraham Lincoln with inpainted artifact

use the primal dual algorithm to solve (14), and remove some artifact from an image (see figure 5). To do so, we provide you with an interactive interface that allows you to select regions of an image that you wish to remove. Once the region is selected, you need to remove the selected pixels from the image, and then apply the inpainting algorithm.

For this task we ask you to use the **weighted TV** primal dual (8). The only difference with the other primal dual algorithms that you have already implemented is that you should add a weight in front of the scalar product. It does not change the prox operators that were previously defined and that you need to use.

Steps of this task.

- Implement a function that takes the image and the position of the pixels you wish to remove and produce an image with the corresponding missing region.
- Implement the **weighted TV** primal dual algorithm to recover the image with the following paramers: $\tau = 1$, $\sigma = 1$, $\lambda = 0.75$. Result should look like figure 6.

Written output for PART 3.

- Provide the reconstructed image from task 1

- Provide the image with the removed artifact from task 2

Hints.

- When you are doing computations on images in matlab such as computing the gradient, it is better to convert them into double, using the command $double(I)$, where I is the image. After you did the computation, if you want to print the images, you need to convert them into the uint8 format, using the command $uint8(I)$.
- In this homework you will work with color images. Inpainting color images is not more complicated than inpainting gray scale images. All you need to do is solve the inpainting problem for each color channel separately. The combination of the three solutions gives you the fully inpainted color image.

References

[1] Antonin Chambolle and Thomas Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *J. Math. Imaging Vis.*, 40(1):120–145, May 2011.