Image Deblurring with Blurred/Noisy Image Pairs

Yohann Salaun

MVA - Introduction à l'imagerie Numérique

– Soutenance de Projet –
25 Janvier 2013

Table of contents

1. Introduction
2. Algorithm Overview
3. Theoretical Description
   - Initialization
   - Kernel Estimation
   - Deconvolution
   - De-ringing
4. Results

Plan

1. Introduction
2. Algorithm Overview
3. Theoretical Description
4. Results

Image deblurring with blurred/noisy image pairs

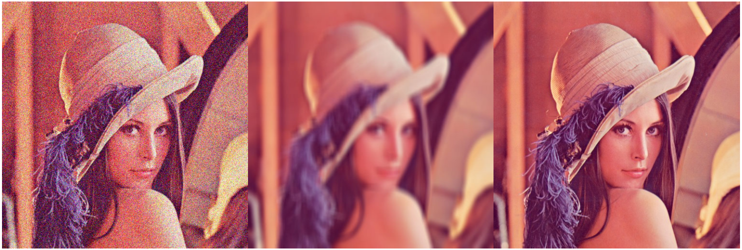

Image deblurring with blurred/noisy image pairs, Lu Yuan, Jian Sun, Long Quan, and Heung-Yeung Shum, Siggraph'07, 2007

Plan

1 Introduction

2 Algorithm Overview

3 Theoretical Description

4 Results

Algorithm Overview

Input : Noisy picture **N**, Blurry picture **B**, estimated kernel size $\mathbf{k}_{size}$
Output : Estimated picture **I**, estimated kernel **k**
$\mathbf{N}_d$= denoise(**N**)
I= $\mathbf{N}_d$
while *change* $> \epsilon$ **do**
- **Estimate** kernel **k** with **I** and **B** s.t. $\mathbf{B} = \mathbf{I} \otimes \mathbf{k}$.
- **Deconvolute** blurred picture **B**.
- **Mix** informations to improve estimation **I**.
- **Compute** *change* between 2 iterations.

end

Plan

1 Introduction

2 Algorithm Overview

3 Theoretical Description
- Initialization
- Kernel Estimation
- Deconvolution
- De-ringing

4 Results

Initialization

Image denoising using scale mixtures of gaussians in the wavelet domain, J. Portilla, V. Strela, M. Wainwright, and E.P. Simoncelli, IEEE, Trans. on Image Processing 12, 11, 1338–1351, 2003.

Kernel Equations

Kernel equation into vector-matrix form:

$$\mathbf{B} = \mathbf{I}\mathbf{k}$$

Kernel Equations

Kernel equation into vector-matrix form:

$$\mathbf{B} = \mathbf{I}\mathbf{k}$$

Tikhonov regularization method:

$$min_{\mathbf{k}\in\mathbb{R}^{\mathbf{k}_{size}}} ||\mathbf{I}\mathbf{k} - \mathbf{B}||_2^2 + \lambda^2||\mathbf{k}||_2^2, \;\; s.t. \;\; \mathbf{k} \in \mathbb{R}^{+\mathbf{k}_{size}} \;\; and \;\; ||\mathbf{k}||_1 = 1$$

Gradient Descent

- $\mathbf{k}^0$ is initialized as the delta function.

Gradient Descent

- $\mathbf{k}^0$ is initialized as the delta function.
- $\mathbf{k}^{n+1} = \mathbf{k}^n + \tau \nabla F(\mathbf{k}^n) = \mathbf{k}^n + 2\tau(\mathbf{I}^T\mathbf{B} - (\mathbf{I}^T\mathbf{I} + \lambda^2 Id)\mathbf{k}^n)$

Gradient Descent

- $\mathbf{k}^0$ is initialized as the delta function.
- $\mathbf{k}^{n+1} = \mathbf{k}^n + \tau \nabla F(\mathbf{k}^n) = \mathbf{k}^n + 2\tau(\mathbf{I}^T\mathbf{B} - (\mathbf{I}^T\mathbf{I} + \lambda^2 Id)\mathbf{k}^n)$
- $\mathbf{k}^{n+1} = Proj_K(\mathbf{k}^{n+1})$

Gradient Descent

- $\mathbf{k}^0$ is initialized as the delta function.
- $\mathbf{k}^{n+1} = \mathbf{k}^n + \tau \nabla F(\mathbf{k}^n) = \mathbf{k}^n + 2\tau(\mathbf{I}^T\mathbf{B} - (\mathbf{I}^T\mathbf{I} + \lambda^2 Id)\mathbf{k}^n)$
- $\mathbf{k}^{n+1} = Proj_K(\mathbf{k}^{n+1})$

with $Proj_K$ computed by :

- turning positive the kernel coefficients $\mathbf{k}_i = max(\mathbf{k}_i, 0)$

Gradient Descent

- $\mathbf{k}^0$ is initialized as the delta function.
- $\mathbf{k}^{n+1} = \mathbf{k}^n + \tau \nabla F(\mathbf{k}^n) = \mathbf{k}^n + 2\tau(\mathbf{I}^T\mathbf{B} - (\mathbf{I}^T\mathbf{I} + \lambda^2 Id)\mathbf{k}^n)$
- $\mathbf{k}^{n+1} = Proj_K(\mathbf{k}^{n+1})$

with $Proj_K$ computed by :

- turning positive the kernel coefficients $\mathbf{k}_i = max(\mathbf{k}_i, 0)$
- normalizing the kernel $\mathbf{k} = \frac{\mathbf{k}}{||\mathbf{k}||}$

Richardson-Lucy Algorithm

Iterative method for deconvolution :

$$\mathbf{I}_i^{k+1} = \mathbf{I}_i^k \sum_j \frac{\mathbf{B}_j}{(\mathbf{k} \otimes \mathbf{I}^k)_j} \mathbf{k}_j[i]$$

where $\mathbf{k}_j[i]$ is the j^{th} coefficients of the kernel $\mathbf{k}$ centered in pixel i.

Residual Deconvolution

Residual instead of pictures to limit ringing artifacts :

- $\Delta\mathbf{I} = \mathbf{I} - \mathbf{N}_d$
- $\Delta\mathbf{B} = \Delta\mathbf{I} \otimes \mathbf{k} = \mathbf{B} - \mathbf{N}_d \otimes \mathbf{k}$

Residual Deconvolution

Residual instead of pictures to limit ringing artifacts :

- $\Delta \mathbf{I} = \mathbf{I} - \mathbf{N}_d$
- $\Delta \mathbf{B} = \Delta \mathbf{I} \otimes \mathbf{k} = \mathbf{B} - \mathbf{N}_d \otimes \mathbf{k}$

Offset added to avoid zero values issues :

$$\Delta \mathbf{I}_i^{k+1} + 1 = (\Delta \mathbf{I}_i^k + 1) \sum_j \frac{\Delta \mathbf{B}_j + 1}{(\mathbf{k} \otimes \Delta \mathbf{I}^k)_j + 1} \mathbf{k}_j[i]$$

Gain-controlled RL deconvolution

$$\mathbf{I}_i^{k+1} = I_{GAIN}[i](\mathbf{I}_i^k \sum_j \frac{\mathbf{B}_j}{(\mathbf{k} \otimes \mathbf{I}^k)_j} \mathbf{k}_j[i])$$

Gain-controlled RL deconvolution

$$\mathbf{I}_i^{k+1} = I_{GAIN}[i](\mathbf{I}_i^k \sum_j \frac{\mathbf{B}_j}{(\mathbf{k} \otimes \mathbf{I}^k)_j} \mathbf{k}_j[i])$$

where I_{GAIN} controls the increase of contrast :

$$I_{GAIN} \sim (1 - \alpha) + \alpha \|\nabla \mathbf{N}_d\|$$

where $\alpha = 0.8$ in the article

Final Deconvolution

- RL-deconvolution on $\mathbf{I}_{RL} = RL(\mathbf{I})$

Final Deconvolution

- RL-deconvolution on $\mathbf{I}_{RL} = RL(\mathbf{I})$
- Gain-controlled RL-deconvolution on $\mathbf{I}_g = RL_{gain}(\mathbf{I})$

Final Deconvolution

- RL-deconvolution on $\mathbf{I}_{RL} = RL(\mathbf{I})$
- Gain-controlled RL-deconvolution on $\mathbf{I}_g = RL_{gain}(\mathbf{I})$
- Compute a detail layer $\mathbf{I}_d = \mathbf{I}_{RL} - F(\mathbf{I}_{RL})$ where F is a low-pass filter such as the bilateral filter

Final Deconvolution

- RL-deconvolution on $\mathbf{I}_{RL} = RL(\mathbf{I})$
- Gain-controlled RL-deconvolution on $\mathbf{I}_g = RL_{gain}(\mathbf{I})$
- Compute a detail layer $\mathbf{I}_d = \mathbf{I}_{RL} - F(\mathbf{I}_{RL})$ where F is a low-pass filter such as the bilateral filter
- Compose the detail layer $\mathbf{I}_d$ and the base layer $\mathbf{I}_g$

Plan

1 Introduction

2 Algorithm Overview

3 Theoretical Description

4 Results

5×5 kernel

9×9 kernel

Different initialization

Kernel size

Article results