Learning PDEs for Image Restoration via Optimal Control

Risheng Liu[1], Zhouchen Lin[2,⋆], Wei Zhang[3], and Zhixun Su[1]

[1] Dalian University of Technology, Dalian 116024, P.R. China
rsliu@mail.dlut.edu.cn, zxsu@dlut.edu.cn
[2] Microsoft Research Asia, Beijing 100190, P.R. China
zhoulin@microsoft.com
[3] The Chinese University of Hong Kong, Shatin, Hong Kong, P.R. China
zw009@ie.cuhk.edu.hk

Abstract. Partial differential equations (PDEs) have been successfully applied to many computer vision and image processing problems. However, designing PDEs requires high mathematical skills and good insight into the problems. In this paper, we show that the design of PDEs could be made easier by borrowing the *learning strategy* from machine learning. In our learning-based PDE (L-PDE) framework for image restoration, there are two terms in our PDE model: (i) a regularizer which encodes the prior knowledge of the image model and (ii) a linear combination of differential invariants, which is data-driven and can effectively adapt to different problems and complex conditions. The L-PDE is learnt from some input/output pairs of training samples via an optimal control technique. The effectiveness of our L-PDE framework for image restoration is demonstrated with two exemplary applications: image denoising and inpainting, where the PDEs are obtained easily and the produced results are comparable to or better than those of traditional PDEs, which were elaborately designed.

1 Introduction

1.1 Prior Work

Partial differential equations (PDEs) have been successfully applied to solve many problems in computer vision and image processing. This kind of methods can date back to the 1960s [1,2]. However, this technique did not draw much attention until the introduction of the concept of scale space by Koenderink [3] and Witkin [4] in the 1980s. The Perona-Malik (P-M) anisotropic equation [5] and the mean curvature motion (MCM) equation [6] further drew great interest from researchers toward designing PDEs for various problems in computer vision and image processing. In general, there are two types of methods for designing PDEs for vision tasks [7]:

1. ***Variational Design***: Basically, variational methods first define an energy functional to collect the desired properties of the output image, including the image prior models (e.g., the Tikhonov regularizer [8] and the total variation (TV) regularizer [9]), and then derive the evolution equations by computing the Euler-Lagrange equation of the energy functional.

⋆ Corresponding author.

K. Daniilidis, P. Maragos, N. Paragios (Eds.): ECCV 2010, Part I, LNCS 6311, pp. 115–128, 2010.

2. ***Direct Design***: Direct methods involve writing down the PDEs directly, based on the mathematical and physical understandings of the problem. This method requires proficiency in the properties of differential operators, in particular nonlinear ones. Famous examples include anisotropic diffusion [5], shock filter [10] and curve evolution [6].

In a geometric view, most traditional PDEs in computer vision and image processing are obtained by either optimizing some global geometric quantities (e.g., length, area, and total squared curvature) or by computing geometric invariants under certain transformation groups. All of these methods require good skills when choosing appropriate PDE forms and predicting the final effect of composing related terms such that the obtained PDEs roughly meet the goals. A lot of trial and error may also be necessary for designing a good PDE. As a result, current methods for designing PDEs greatly limit the applications of PDEs to wider and more complex scopes. This motivates us to explore whether we can acquire PDEs that are more powerful but require much less human effort.

1.2 Our Approach

Inspired by learning-based methods in machine learning, we would like to explore a framework for learning PDEs to accomplish various computer vision and image processing tasks. In this paper, as preliminary work, we propose a learning-based PDE (L-PDE) framework for image restoration. For image restoration problems, we know that the output image should obey some statistical models of natural images. Such statistical models can serve as the regularizer term in our PDE, which controls the output image, making it a natural image. Hence this term is called the *regularization* term. The other term in our PDE is to cope with different image restoration problems and different data. As most image restoration problems are translationally and rotationally invariant, i.e., when the input image is translated or rotated by some amount the output image is also translated or rotated by the same amount, this second term must be functions of fundamental differential invariants [11] that are invariant under translation and rotation. We assume that the second term is a linear combination of the fundamental differential invariants. Although a linear combination is simple, our PDE model is already general enough and many existing PDEs can be viewed as a special case of our model. The linear combination coefficients are learnt from real data so that the learnt PDE can adapt to different image restoration problems and different data. Hence the second term is called the *data-driven differential invariant* term.

To learn the coupling coefficients among the differential invariants in the data-driven term, we prepare some input/output training image pairs and adopt a technique called PDE-based optimal control [12]. Once the coefficients are computed, the L-PDE is obtained and can be applied to test images. Hence with our framework, the most effort on obtaining a PDE is preparing some input/output training image pairs. So our L-PDE framework might be a possible way of designing PDEs for vision tasks in a lazy manner. Though the optimal control technique has already been applied to some computer vision problems, such as optical flow estimation [13] and tracking [14], we use it in a different way. We aim at determining the form (coefficients) of the PDEs, while the existing

work uses the optimal control to determine the outputs of their PDEs, which are known *apriori*. In short, *our L-PDE framework connects PDE-based methods and learning-based methods via optimal control.*

2 Learning-Based PDE Model

In this section, we present the form of the PDEs in our L-PDE framework for image restoration. We denote f as the input image and u as the desired output image. The meaning of the notations that will be used hereafter can be found in Table 1.

Table 1. Notations

Ω	An open bounded region of $\mathbb{R}^2$	$\partial\Omega$	Boundary of Ω
(x, y)	$(x, y) \in \Omega$, spatial variable	t	$t \in (0, T_f)$, temporal variable
Q	$\Omega \times (0, T_f)$	Γ	$\partial\Omega \times (0, T_f)$
$\|\cdot\|$	L^2 norm	∇u	Gradient of u
$\mathbf{H}_u$	Hessian of u	$\mathrm{div}(\mathbf{u})$	Divergence of $\mathbf{u}$
$\wp$	$\wp = \{(0,0),(0,1),(1,0),(0,2),(1,1),(2,0)\}$, index set for differentiation		
$\kappa(u)$	$\kappa(u) = \mathrm{div}\left(\frac{\nabla u}{\|\nabla u\|}\right)$, mean curvature of u		

2.1 Description of Our PDE Model

Our PDE model is an evolutionary PDE combining a TV regularizer and a linear combination of fundamental differential invariants:

$$\begin{cases} \frac{\partial u}{\partial t} = L(u, \mathbf{a}), & (x, y, t) \in Q, \\ u = 0, & (x, y, t) \in \Gamma, \\ u|_{t=0} = f, & (x, y) \in \Omega, \end{cases} \tag{1}$$

where $L(u, \mathbf{a}) = \kappa(u) + F(u, \mathbf{a})$. The Dirichlet boundary condition[1] is for ease of mathematical deduction. The forms and the geometric meanings of $\kappa(u)$ and $F(u, \mathbf{a})$ will be presented below.

Total Variation Regularization Term: The TV regularization has been successfully incorporated in PDEs for a large class of computer vision and image processing problems due to its mathematical tractability and effectiveness in representing the statistical model of natural images. It was first introduced to computer vision and image processing by Rudin, Osher and Fatemi (ROF) in their paper on edge preserving image denoising [9]. It first defines a variational minimization model $\min_u \int_\Omega \|\nabla u\| d\Omega$ in the bounded variation space (which allows for piecewise constant images) and then derives the mean curvature $\kappa(u)$ as the regularization term in its associated Euler-Lagrange equation [9,7]. The TV regularization is especially useful in applications, e.g., image restoration, where edges are to be respected. That is why our PDE model incorporates $\kappa(u)$.

[1] As in real applications f will be padded with zeros of several pixels' width around the input image, the difference between the Dirichlet boundary condition in our model and the Neumann boundary condition in traditional PDEs is slight.

Table 2. The fundamental differential invariants up to the second order

$\mathbf{inv}(u) = [\text{inv}_0(u), ..., \text{inv}_5(u)]^T$		
i	$\text{inv}_i(u)$	
0	f	
1	u	Zeroth Order
2	$\|\nabla u\|^2 = u_x^2 + u_y^2$	First Order
3	$\text{tr}(\mathbf{H}_u) = u_{xx} + u_{yy}$	Second Order
4	$\text{tr}(\mathbf{H}_u^2) = u_{xx}^2 + 2u_{xy}^2 + u_{yy}^2$	
5	$(\nabla u)^T \mathbf{H}_u (\nabla u) = u_x^2 u_{xx} + 2u_x u_y u_{xy} + u_y^2 u_{yy}$	

Data-Driven Differential Invariant Term: As we have explained in Section 1.2, the data-driven differential invariant term is a linear combination of fundamental differential invariants that are invariant under translation and rotation. For 2D scalar images, there are five such fundamental differential invariants up to the second order [11]. They are listed in Table 2 (f is added as the zeroth invariant due to the following geometric considerations). All the differential invariants have geometric meanings. $\text{inv}_0(u) = f$ is the input image. $\text{inv}_1(u) = u$ is the desired output image. $\text{inv}_2(u) = \|\nabla u\|^2$ is the squared norm of the gradient. $\text{inv}_3(u) = \text{tr}(\mathbf{H}_u)$ is the Laplacian, which has been widely used to measure the smoothness of an image [15]. $\text{inv}_4(u) = \text{tr}(\mathbf{H}_u^2)$, known as "deviation from flatness", is another useful way to measure the local "unflatness" of the image. $\text{inv}_5(u) = (\nabla u)^T \mathbf{H}_u (\nabla u)$ is a kind of image "curvature", which has been used as a general purpose visual front-end operation [16]. Using such differential invariants, all local intrinsic properties of images, which should be invariant to coordinate transformation, can be described. Therefore, the data-driven term for our L-PDE model can be written as:

$$F(u, \mathbf{a}) = \mathbf{a}(t)^T \mathbf{inv}(u), \tag{2}$$

where $\mathbf{a}(t) = [a_0(t), ..., a_5(t)]^T$ are coefficient functions, which are used to control the evolution of u. For different problems, $\mathbf{a}(t)$ is different. They are learnt from training images and hence our L-PDE can adapt to different problems and data. We will present a PDE-based optimal control framework to learn these coefficient functions in Section 3.

2.2 Connection between L-PDE and Traditional PDEs

In this subsection, we discuss the relationship between our L-PDE model and some well-known related work.

Traditional PDEs were designed with different insights. However, as shown in Table 3, many of those for image restoration in fact share a common formulation and are all special cases of our proposed L-PDE model. The difference between these PDEs lies in the choice of $\mathbf{a}(t)$ and the regularization term. However, our L-PDE model and the traditional PDEs present intrinsically different perspectives on interpreting the form of PDEs. Traditional PDEs are all crafted by people with skills, based on their insight to the problems, whereas our model automatically determines the PDEs from real data. One can easily see that manually designed PDEs only correspond to trivial coefficient functions, where only popular differential invariants, e.g., the Laplacian and the zeroth

Table 3. Reformulating some popular PDEs in our L-PDE model

PDE	$\mathbf{a}(t)$ in data-driven term	Regularization term
Gaussian scale space [3]	$\mathbf{a}(t) = [0, 0, 0, 1, 0, 0]^T$	–
Tikhonov [8]	$\mathbf{a}(t) = [1, -1, 0, 1, 0, 0]^T$	–
ROF [9], TV inpainting [17]	$\mathbf{a}(t) = [1, -1, 0, 0, 0, 0]^T$	$\kappa(u)$

order invariants, are used. Moreover, the nonzero coefficients are also *special constants*. In comparison, the coefficients in our L-PDEs can be much more flexible. They may not be sparse. They can be arbitrary real numbers and can even vary with time. So our L-PDE model can be much more adaptive to the input images and solve different image restoration problems in a unified framework.

3 Learning Coefficients via Optimal Control

3.1 The Objective Functional

Given the form of the general data-driven term in (2), we have to determine the coefficient functions $\mathbf{a}(t)$ in order to obtain a workable PDE. We may prepare some pairs of input/output training samples $(f_k, \tilde{u}_k)$, where f_k is the input image and $\tilde{u}_k$ is the expected output image. Since the final output of our PDE should be close to the ground truth, the coefficient functions should minimize the following functional:

$$J(\{u_k\}_{k=1}^K, \mathbf{a}) = \frac{1}{2}\sum_{k=1}^{K}\int_{\Omega}(u_k(T_f) - \tilde{u}_k)^2 d\Omega + \frac{1}{2}\sum_{i=0}^{5}\alpha_i \int_0^{T_f} a_i^2(t)dt, \qquad (3)$$

where $u_k(T_f)$ is the output image at time $t = T_f$[2] computed from (1) when the input image is f_k, and α_i are positive weighting parameters[3]. The first term of J requires the final output of our PDE to be close to the ground truth. The second term is for regularization so that this optimal control problem is well-posed.

3.2 Solving the Optimal Control Problem

Then we have the following optimal control problem with PDE constraints:

$$\min_{\mathbf{a}} J(\{u_k\}_{k=1}^K, \mathbf{a}), \quad s.t. \begin{cases} \frac{\partial u_k}{\partial t} = L(u_k, \mathbf{a}), & (x, y, t) \in Q, \\ u_k = 0, & (x, y, t) \in \Gamma, \\ u_k|_{t=0} = f_k, & (x, y) \in \Omega. \end{cases} \qquad (4)$$

By introducing the adjoint equation of (4), the Gâteaux derivative of J can be computed and consequently, the (locally) optimal $\mathbf{a}(t)$ can be computed via gradient based

[2] For different problems, T_f may be different. How to determine the optimal T_f is left to future work.

[3] In this paper, we simply fix $\alpha_i = 10^{-7}$, $i = 0, ..., 5$.

algorithms (e.g., conjugate gradient [18]). Here, we give the adjoint equation and Gâteaux derivative directly due to the page limit[4].

Adjoint Equation: The adjoint equation of (4) is:

$$\begin{cases} \frac{\partial \varphi_k}{\partial t} + \sum\limits_{(p,q)\in\wp} (-1)^{(p+q)} \frac{\partial^{p+q}(\sigma_{pq}(u_k)\varphi_k)}{\partial x^p \partial y^q} = 0, & (x,y,t) \in Q, \\ \varphi_k = 0, & (x,y,t) \in \Gamma, \\ \varphi_k|_{t=T_f} = \tilde{u}_k - u_k(T_f), & (x,y) \in \Omega, \end{cases} \tag{5}$$

where

$$\sigma_{pq}(u) = \frac{\partial L(u)}{\partial u_{pq}} = \frac{\partial \kappa(u)}{\partial u_{pq}} + \sum_{i=0}^{5} a_i \frac{\partial \text{inv}_i(u)}{\partial u_{pq}} \quad \text{and} \quad u_{pq} = \frac{\partial^{p+q} u}{\partial x^p \partial y^q}.$$

Gâteaux Derivative of the Functional: With the help of the adjoint equation, at each iteration the derivative of J with respect to $\mathbf{a}(t)$ is as follows:

$$\frac{\partial J}{\partial a_i} = \alpha_i a_i - \sum_{k=1}^{K} \int_{\Omega} \varphi_k \text{inv}_i(u_k) d\Omega, \quad i = 0, ..., 5. \tag{6}$$

where the adjoint function φ_k is the solution to (5).

3.3 Initialization of $\mathbf{a}(t)$

A good initialization of $\mathbf{a}(t)$ results in a better approximation power of the learnt PDE and also makes the optimization process shorter. Here we propose a heuristic method for initializing the coefficient functions. At each time step, $\frac{\partial u_k(t)}{\partial t}$ is expected to be $\frac{\tilde{u}_k - u_k(t)}{T_f - t}$ so that u_k tends to the expected output $\tilde{u}_k$. On the other hand, with $\frac{\partial u_k(t)}{\partial t} = L(u_k, \mathbf{a})$, we want $\mathbf{a}(t)$ to minimize:

$$\sum_{k=1}^{K} \int_{\Omega} \left(L(u_k, \mathbf{a}) - \frac{\partial u_k(t)}{\partial t} \right)^2 d\Omega = \sum_{k=1}^{K} \int_{\Omega} [\mathbf{p}_k(t)^T \mathbf{a}(t) - d_k(t)]^2 d\Omega, \tag{7}$$

where $\mathbf{p}_k(t) = \mathbf{inv}(u_k)$ and $d_k(t) = \frac{\tilde{u}_k - u_k(t)}{T_f - t} - \kappa(u_k)$. So the initial $\mathbf{a}(t)$ can be obtained by solving the following system[5]:

$$\mathbf{P}(t)\mathbf{a}(t) = \mathbf{d}(t), \tag{8}$$

where $\mathbf{P}(t) = \sum\limits_{k=1}^{K} \int_{\Omega} \mathbf{p}_k(t)\mathbf{p}_k(t)^T d\Omega$ and $\mathbf{d}(t) = \sum\limits_{k=1}^{K} \int_{\Omega} \mathbf{p}_k(t) d_k(t) d\Omega$.

[4] For more details and a more mathematically rigorous exposition, please see Supplementary Material and refer to [19,20,21].

[5] For notational convenience, we simply write integrals here. In real computation, the integrals should be discretized.

4 Our L-PDE Framework for Image Restoration

We now summarize our L-PDE framework for image restoration in Algorithm 1. After the PDE is learnt, it can be applied to new test images by solving (1), whose input f is the test image and the solution $u(T_f)$ is the desired output image.

Algorithm 1. (The framework to learn PDEs for image restoration)

Require: Training image pairs $(f_k, \tilde{u}_k)$, $k = 1, ..., K$; T_f.
1: Initialize $\mathbf{a}(t)$, $t \in [0, T_f)$, by solving (8).
2: **while** not converged **do**
3: Compute $\frac{\partial J}{\partial a_i}$, $i = 0, ..., 5$, using (6).
4: Decide the search direction using the conjugate gradient method [18];
5: Perform golden search along the search direction and update $\mathbf{a}(t)$.
6: **end while**
Ensure: The coefficient functions $\mathbf{a}(t)$, $t \in [0, T_f)$.

5 Experimental Results

In this section, we demonstrate the applications of our L-PDE framework for image restoration to two problems, denoising and inpainting. Our experiments are done on grayscale images. *For the best visual comparison, the readers are encouraged to inspect the images in this section on screen.*

5.1 Image Denoising

For the image denoising problem, we compare our learnt PDE to the state-of-the-art PDE denoising methods, P-M [5], ROF [9] and TV-L^1 [22], on images with both synthetic and real noise. For each experiment, 6 noisy images and their ground truths are randomly chosen to train the coefficients in the L-PDE, and the remaining images are the test images. The parameters in the three compared PDEs are tuned to so that the mean peak signal to noise ratio (PSNR) of all test images are the highest.

Denoising Images with Synthetic Noise. We perform two simulation experiments on images with synthetic noise. The images are chosen from the Berkeley image database [23]. There are 86 images in total[6] and the image size is 321×481 pixels. For the first experiment, zero-mean Gaussian white noise with $\sigma = 25$ is added to the images. For the second experiment, a mixture of zero-mean Gaussian white noise ($\sigma = 50$), Poisson noise (λ being the pixel values) and the salt & pepper noise ($d = 0.1$) is added to the images. For both experiments, T_f is chosen as 2.

Fig. 1 compares the results of the L-PDE with those of the traditional PDEs on images with Gaussian noise. It shows that the L-PDE preserves details better than the traditional PDEs. Moreover, the L-PDE also achieves higher PSNRs. Fig. 2 shows the comparison of denoising results on mixture noise. One can see that the P-M model cannot remove the salt & pepper noise well. Although ROF and TV-L^1 perform better than

[6] We randomly choose 6 images for training and the remaining 80 images for testing.

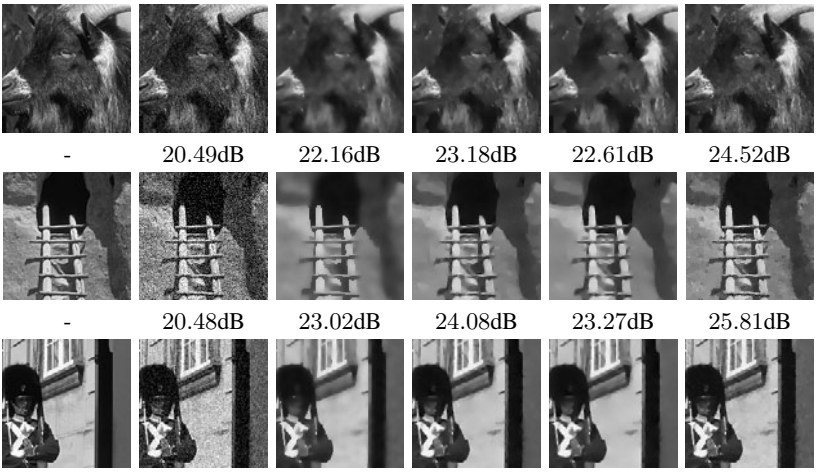


Fig. 1. The results of denoising images with Gaussian noise. (a) Original noiseless image. (b) Noisy image with additive Gaussian noise ($\sigma = 25$). (c)-(f) Denoised images using the P-M, ROF, TV-L^1, and our L-PDE models, respectively. The PSNRs are presented below each image.

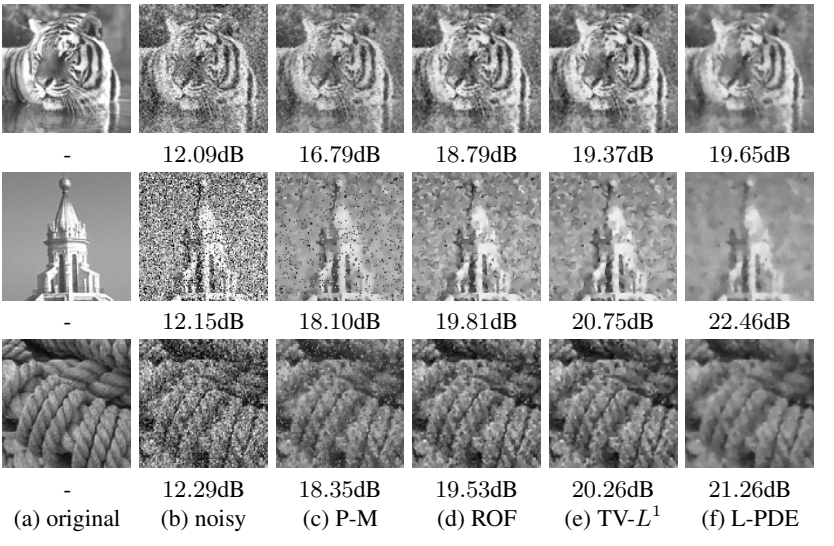


Fig. 2. The results of denoising images with mixture noise. (a) Original noiseless image. (b) Noisy image with mixture noise. (c)-(f) Denoised images using the P-M, ROF, TV-L^1, and our L-PDE models, respectively. The PSNRs are shown below each image.

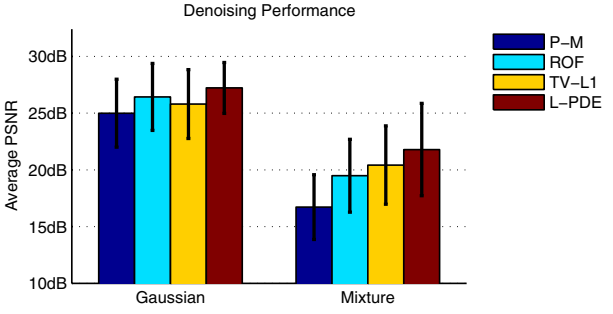


Fig. 3. Performance of denoising as measured in PSNR. In each experiment, the average PSNR (colored bar) and standard deviation (thick vertical line) of the denoised images in the test set is shown.

P-M, their denoised images remain noisy. In comparison, our L-PDE suppresses almost all of the noise while preserving the details well.

The quantitative results of the experiments on two kinds of noise are summarized in Fig. 3. One can see that none of the three traditional PDEs can work well on both kinds of noise. On Gaussian noise, ROF outperforms P-M and TV-L^1 and has comparable results with L-PDE, because this model is specifically designed for Gaussian noise. However, ROF does not work well on mixture noise. On mixture noise, the performance of TV-L^1 is better than ROF and P-M. This is because TV-L^1 incorporates a contrast invariant fidelity term, which makes it more adaptive to unknown noise than ROF and P-M. So the performance of the traditional PDEs heavily depends on the test data. In contrast, our L-PDE outperforms all the compared traditional PDEs in both denoising experiments. This is because our L-PDE is data-driven. It learns the form of the PDE from training data to fit the noise, no matter whether the noise distribution is known or unknown.

Denoising Images with Really Unknown Noise. To further testify to the data-driven nature of our L-PDE, in this experiment we test on images with really unknown noise. We take 240 images, each with a size 300×300 pixels, of 8 objects using a Canon 30D digital camera, setting its ISO to 1600. For each object, 30 images are taken without changing the camera settings (by fixing the focus, aperture and exposure time) and without moving the camera position. The mean image of them can be regarded as the noiseless ground truth image. We randomly choose 6 objects. For each object we randomly choose one noisy image. These noisy images and their ground truth images are used to train an L-PDE, where T_f is set as 1. Then we compare our L-PDE with the traditional PDEs on images of the remaining 2 objects. In Fig. 4, we show the comparison of these results. The zoomed-in regions show that the output of the L-PDE has less severe artifacts and is sharper than that of other algorithms. As shown in Table 4, the PSNRs of our L-PDE are dramatically higher than those of traditional PDEs. This shows that our L-PDE framework can easily adapt to different types of noise and obtain

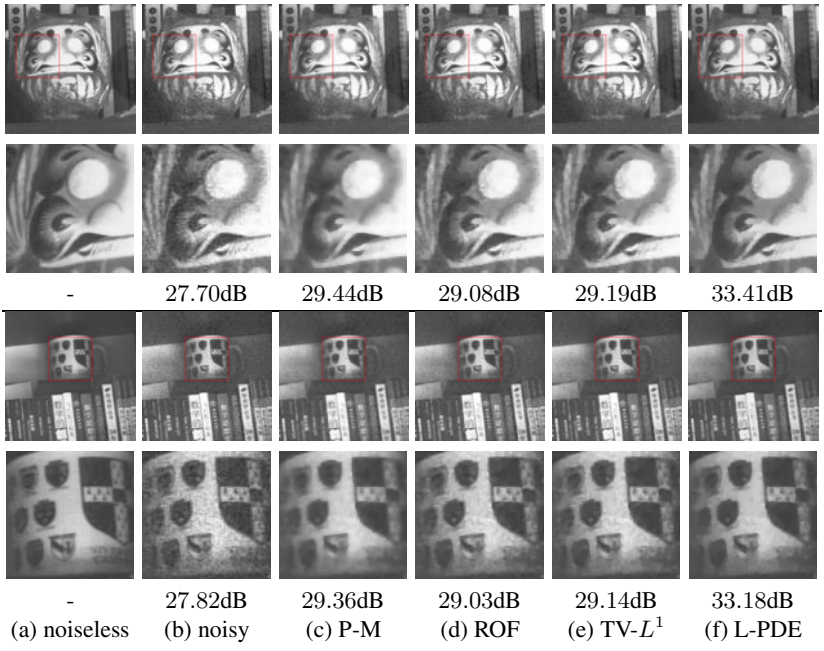


Fig. 4. The results of denoising images with really unknown noise. The second and fourth rows show the corresponding zoomed-in regions in the boxes in the first and third rows, respectively. (a) The estimated noiseless image. (b) Captured noisy image. ((c)-(f) Denoised images using the P-M, ROF, TV-L^1, and our L-PDE models, respectively. The estimated PSNRs are shown below each image.

Table 4. Denoising results (in PSNR, presented in "mean $\pm$ std-dev dB") of the images of the remaining two objects, each object having 30 noisy images

Object	Noisy	P-M	ROF	TV-L^1	L-PDE
1	27.97 ± 0.19dB	29.55 ± 0.28dB	29.22 ± 0.26dB	29.34 ± 0.27dB	33.25 ± 0.10dB
2	28.01 ± 0.31dB	29.89 ± 0.48dB	29.50 ± 0.44dB	29.63 ± 0.45dB	33.36 ± 0.09dB

L-PDEs that fit for different types of noise well. In contrast, as the traditional PDEs were designed under specific assumptions on the types of noise, they may not fit for other types of noise as well as our L-PDEs.

5.2 Image Inpainting

In this subsection, we apply our L-PDE framework to the image inpainting problem. Obviously, to obtain a "good" inpainting result, proper information of the image and the structure of the missing pixels are needed to impose certain priors on the solution. Different from the TV inpainting model [17], which *only* propagates $\kappa(u)$ to fill in the missing region R, our L-PDE learns the structure of the missing pixels in R from the training data and applies both $\kappa(u)$ and the data-driven term to the test image. As the

data inside the region R of missing pixels is unavailable, we cannot involve the input image f, which is $\text{inv}_0(u)$, in our L-PDE model. So we limit the coefficient $a_0(t)$ to be 0 throughout the optimal control process. In this experiment, $T_f = 4$.

Fig. 5 shows a typical result of noisy image inpainting. We use 6 noisy images (masked by dense text) with their ground truths to train an L-PDE and then apply it to test images. Comparing to the FoE inpainting model [24], which is not a PDE-based method, both TV inpainting [17] and our L-PDE can simultaneously denoise the image and fill in the missing pixels. Moreover, the visual quality and PSNR of our L-PDE are both better than those of TV inpainting [17]. We also apply this L-PDE to other images with purely random masks. Fig. 6 shows that the proposed method also works well.

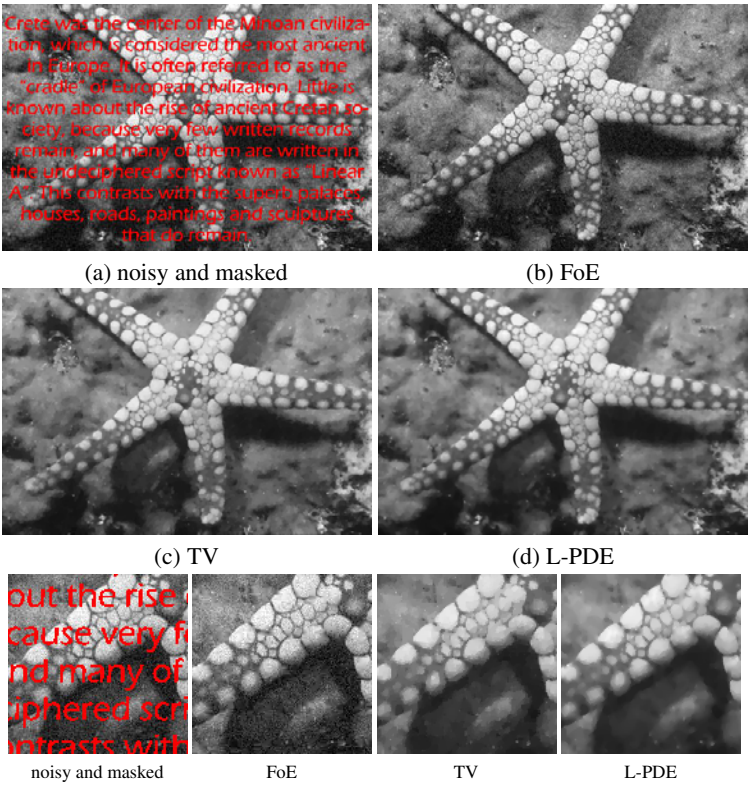


Fig. 5. The results of noisy image inpainting. Gaussian noise with $\sigma = 15$ is added and then texts are overlaid. PSNRs are computed on the whole image. (a) Noisy image with overlaid text; PSNR = 14.29dB. (b) Inpainting result from FoE; PSNR = 24.42dB. (c) Inpainting result from TV; PSNR = 26.84dB. (d) Inpainting result from L-PDE; PSNR = 27.68dB. (e) Close-up comparison of these algorithms.

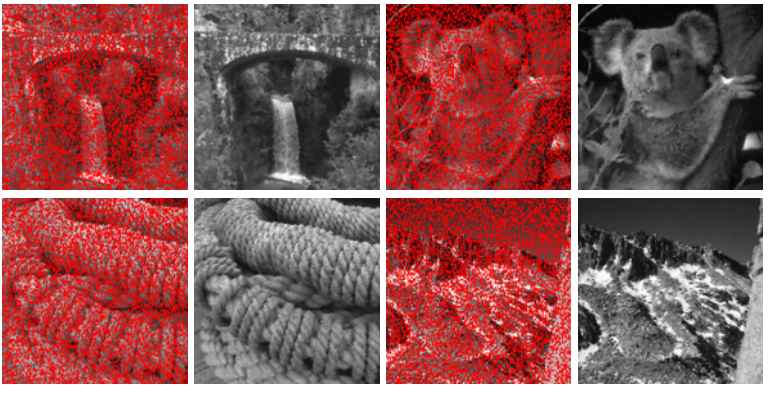
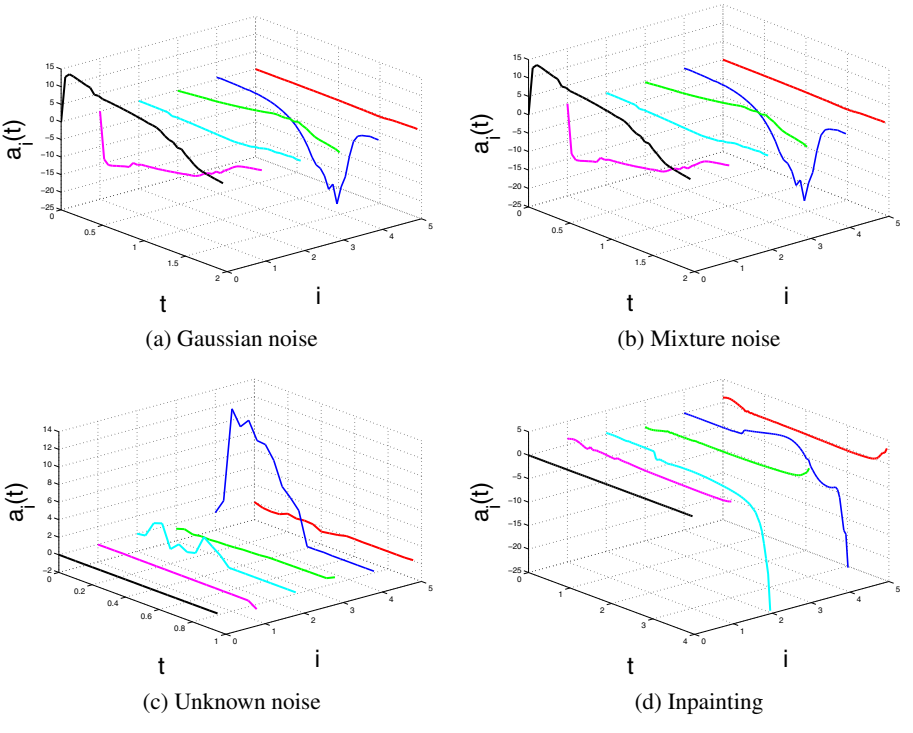


Fig. 6. The results of purely randomly masked image inpainting (50% pixels are masked), using our L-PDE. The first and the third columns show the masked images. The second and fourth columns show the corresponding inpainted images.



(a) Gaussian noise (b) Mixture noise

(c) Unknown noise (d) Inpainting

Fig. 7. Learnt coefficients $a_i(t), i = 0, 1, ..., 5$, of PDEs for different image restoration problems.

Finally, we show the curves of the learnt coefficients of PDEs for different image restoration problems in Figure 7. Currently we are unable to analyze the obtained PDEs in depth as this work seems to be non-trivial. So we leave the analysis to future work.

6 Conclusions and Future Work

In this paper, we have presented a framework of learning PDEs from training data for image restoration. The experiments on natural image denoising and inpainting show that our framework is effective. Compared to the traditional PDEs, our L-PDEs are obtained much more easily. In the future, we would like to improve and enrich our work in the following aspects. First, solve the theoretical issues in our L-PDE model, e.g., the existence and uniqueness of the solution to (1). Second, develop more efficient numerical algorithms to solve our optimal control problem (4). Third, we will also consider incorporating the idea of diffusion tensor [25] and generalizing our framework for vector/matrix/tensor valued images. Finally, we will also apply our framework to more computer vision and image processing problems.

References

1. Gabor, D.: Information theory in electron microscopy. Laboratory Investigation 14, 801–807 (1965)
2. Jain, A.: Partial differential equations and finite-difference methods in image processing, part 1. Journal of Optimization Theory and Applications 23, 65–91 (1977)
3. Koenderink, J.: The structure of images. Biological Cybernetics 50, 363–370 (1984)
4. Witkin, A.: Scale-space filtering. In: International Joint Conference on Artificial Intelligence, IJCAI (1983)
5. Pietro, P., Jitendra, M.: Scale-space and edge detection using anisotropic diffusion. IEEE Transactions on Pattern Analysis and Machine Intelligence 12, 629–639 (1990)
6. Alvarez, L., Lions, P.L., Morel, J.M.: Image selective smoothing and edge detection by nonlinear diffusion. SIAM Journal on Numerical Analysis 29, 845–866 (1992)
7. Chen, T., Shen, J.: Image processing and analysis: variational, PDE, wavelet, and stochastic methods. SIAM Publisher, Philadelphia (2005)
8. Tikhonov, A., Arsenin, V.: Solutions of ill-posed problems. Halsted Press (1977)
9. Rudin, L., Osher, S., Fatemi, E.: Nonlinear total variation based noise removal algorithms. Physica D 60, 259–268 (1992)
10. Osher, S., Rudin, L.: Feature-oriented image enhancement using shock filters. SIAM Journal on Numerical Analysis 27, 919–940 (1990)
11. Olver, P.: Applications of Lie groups to differential equations. Springer, Heidelberg (1993)
12. Hinze, M., Pinnau, R., Ulbrich, M., Ulbrich, S.: Optimization with PDE contraints. Springer, Heidelberg (2009)
13. Papadakis, N., Corpetti, T., Memin, E.: Dynamically consistent optical flow estimation. In: International Conference on Computer Vision, ICCV (2007)
14. Papadakis, N., Memin, E.: Variational optimal control technique for the tracking of deformable objects. In: International Conference on Computer Vision, ICCV (2007)
15. Florack, L., Romeny, B., Koenderink, J., Viergever, M.: Scale and the differential structure of image. Image and Vision Computing 10, 376–388 (1992)

16. Lindeberg, T.: Discrete derivative approximations with scale-space properties: a basis for low-level feature extraxtion. Journal of Mathematical Imaging and Vision 3, 349–376 (1993)
17. Chan, T., Shen, J.: Mathematical models for local nontexture inpaintings. SIAM Journal on Applied Mathematics 62, 1019–1043 (2002)
18. Stoer, J., Bulirsch, R.: Introduction to numerical analysis, 2nd edn. Springer, Heidelberg (1998)
19. Lions, J.: Optimal control systems governed by partial differential equations. Springer, Heidelberg (1971)
20. Lin, Z., Zhang, W., Tang, X.: Learning partial differential equations for computer vision. Technical report, Microsoft Research, MSR-TR-2008-189 (2008)
21. Lin, Z., Zhang, W., Tang, X.: Designing partial differential equations for image processing by combining differental invariants. Technical report, Microsoft Research, MSR-TR-2009-192 (2009)
22. Chan, T., Esedoglu, S.: Aspects of total variation regularizaed L^1 function approximation. SIAM Journal on Applied Mathematics 65, 1817–1837 (2005)
23. Martin, D.R., Fowlkes, C., Tal, D., Malik, J.: A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In: International Conference on Computer Vision, ICCV (2001)
24. Roth, S., Black, M.J.: Fields of experts: a framework for learning image priors. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR (2005)
25. Weickert, J., Hagen, H.: Visualization and processing of tensor fields. Springer, Heidelberg (2009)