Exercise 5. Rigid Transform Blending and Variational Methods

Dongho Kang

16-948-598

May 21, 2017

MATLAB R2016b version was used for coding and testing:

MathWorks, MATLAB R2016b (9.1.0.441655)
64-bit (maci64)

The ***code*** directory contains the followings:

part2 script .m file for exercise part 2.

lotr.jpg original image file.

results directory contains result images of part 2.

For running ***part2.m*** script, adjust options first:

gf_on true for Gaussian filtering on.

hd_on true for Heat diffusion on.

va_on true for variational method on.

save_jpg true for saving result images.

results_dir result images directory path.

Note that **these script was only tested in MATLAB R2016b environment**.

1 EXERCISE PART 1: UNDERSTANDING AND UTILIZING DUAL QUATERNION

1.1 TASK 1: THINKING ABOUT FUNDAMENTAL PROPERTIES

- How do dual quaternions represent rotations and translations?

  Unit dual quaternions naturally represent 3D rotation, when the dual part $\mathbf{q}_\epsilon = 0$ (thus $\hat{\mathbf{q}} = \mathbf{q_0} + \epsilon\mathbf{q}_\epsilon = \mathbf{q_0}$). Dual quaternion multiplication with unit dual quaternion $\hat{\mathbf{t}} = 1 + \frac{\epsilon}{2}(t_0 i + t_1 j + t_2 k)$ corresponds to translation by vector (t_0, t_1, t_2) represent 3D translation. [1]

- What is the advantage of representing rigid transformations with dual quaternions for blending?

  The linear combination of dual quaternions does not make artifacts or skin-collapsing effect, thus blending using dual quaternions is fast and more robust than using homogeneous matrix. Moreover, since dual quaternions require only 8 floats per transformation, instead of the 12 required by matrices, they are more memory efficient. [1]

- Briefly explain one fundamental disadvantage of using quaternion based shortest path blending for rotations as compared to linear blend skinning?

  Dual quaternions causes "flipping artifacts" which occurs with joint rotations of more than 180 degrees. Because of its shortest path property, when the other path becomes shorter, the skin changes its shape discontinuously. [1]

1.2 TASK 2: DERIVATIONS AND DEEPER UNDERSTANDING

- For a dual quaternion $\hat{\mathbf{q}} = \cos(\hat{\theta}/2) + \hat{\mathbf{s}}\sin(\hat{\theta}/2)$, prove that $\hat{\mathbf{q}}^t = \cos(t\hat{\theta}/2) + \hat{\mathbf{s}}\sin(t\hat{\theta}/2)$

  Starting with $\hat{\mathbf{q}}$:

$$\hat{\mathbf{q}} = \cos(\hat{\theta}/2) + \hat{\mathbf{s}}\sin(\hat{\theta}/2) \tag{1.1}$$

  As $\hat{\mathbf{q}}^t = \exp\big(t\log(\hat{\mathbf{q}})\big)$,

$$\begin{aligned}\hat{\mathbf{q}}^t &= \exp\big(t\log(\hat{\mathbf{q}})\big) &(1.2)\\ &= \exp\Big(t\log\big(\cos(\hat{\theta}/2) + \hat{\mathbf{s}}\sin(\hat{\theta}/2)\big)\Big) &(1.3)\end{aligned}$$

Plug in $\log\big(\cos(\hat{\theta}/2)+\hat{\mathbf{s}}\sin(\hat{\theta}/2)\big)=\hat{\mathbf{s}}\frac{\hat{\theta}}{2}$ to equation 1.3:

$$\hat{\mathbf{q}}^t=\exp\Big(\frac{t\hat{\theta}}{2}\hat{\mathbf{s}}\Big) \tag{1.4}$$

Let $\hat{\mathbf{a}}=\frac{t\hat{\theta}}{2}\hat{\mathbf{s}}$. Since $\hat{\theta}=\theta_0+\epsilon\theta_\epsilon$ and $\hat{\mathbf{s}}=\mathbf{s}_0+\epsilon\mathbf{s}_\epsilon$.

$$\begin{aligned}\hat{\mathbf{a}}&=\frac{t\hat{\theta}}{2}\hat{\mathbf{s}} &(1.5)\\ &=\frac{t}{2}(\theta_0+\epsilon\theta_\epsilon)(\mathbf{s}_0+\epsilon\mathbf{s}_\epsilon) &(1.6)\\ &=\frac{t}{2}(\theta_0\mathbf{s}_0+\epsilon\theta_0\mathbf{s}_\epsilon+\epsilon\theta_\epsilon\mathbf{s}_0+\cancelto{0}{\epsilon^2\theta_\epsilon\mathbf{s}_\epsilon}) &(1.7)\\ &=\underbrace{\Big(\frac{t}{2}\theta_0\mathbf{s}_0\Big)}_{:=\mathbf{a}_0}+\epsilon\underbrace{\Big(\frac{t}{2}\theta_0\mathbf{s}_\epsilon+\frac{t}{2}\theta_\epsilon\mathbf{s}_0\Big)}_{:=\mathbf{a}_\epsilon} &(1.8)\end{aligned}$$

Since exponential of dual quaternion is given by $e^{\hat{\mathbf{q}}}=\cos(\|\hat{\mathbf{q}}\|)+\frac{\hat{\mathbf{q}}}{\|\hat{\mathbf{q}}\|}\sin(\|\hat{\mathbf{q}}\|)$, then $e^{\hat{\mathbf{a}}}=\cos(\|\hat{\mathbf{a}}\|)+\frac{\hat{\mathbf{a}}}{\|\hat{\mathbf{a}}\|}\sin(\|\hat{\mathbf{a}}\|)$. Here, the norm of dual quaternion $\hat{\mathbf{a}}$ is $\|\hat{\mathbf{a}}\|=\|\mathbf{a}_0\|+\epsilon\frac{<\mathbf{a}_0,\mathbf{a}_\epsilon>}{\|\mathbf{a}_0\|}$. Looking into $<\mathbf{a}_0,\mathbf{a}_\epsilon>$ and $\|\mathbf{a}_0\|$,

$$\begin{aligned}<\mathbf{a}_0,\mathbf{a}_\epsilon>&=<\frac{t}{2}\theta_0\mathbf{s}_0,\frac{t}{2}\theta_0\mathbf{s}_\epsilon+\frac{t}{2}\theta_\epsilon\mathbf{s}_0> &(1.9)\\ &=<\frac{t}{2}\theta_0\mathbf{s}_0,\frac{t}{2}\theta_0\mathbf{s}_\epsilon>+<\frac{t}{2}\theta_0\mathbf{s}_0,\frac{t}{2}\theta_\epsilon\mathbf{s}_0> &(1.10)\end{aligned}$$

$$\begin{aligned}\|\mathbf{a}_0\|&=\sqrt{<\mathbf{a}_0,\mathbf{a}_0>} &(1.11)\\ &=\sqrt{<\frac{t}{2}\theta_0\mathbf{s}_0,\frac{t}{2}\theta_0\mathbf{s}_0>} &(1.12)\end{aligned}$$

Note that $<\mathbf{s}_0,\mathbf{s}_0>=1$ and $<\mathbf{s}_0,\mathbf{s}_\epsilon>=0$. Thus equation 1.10 and 1.12 can be expressed as follows:

$$<\mathbf{a}_0,\mathbf{a}_\epsilon>=\Big(\frac{t}{2}\Big)^2\theta_0\theta_\epsilon \tag{1.13}$$

$$\|\mathbf{a}_0\|=\sqrt{\Big(\frac{t}{2}\theta_0\Big)^2}=\frac{t}{2}\theta_0 \tag{1.14}$$

Plug 1.13 and 1.14 into $\|\hat{\mathbf{a}}\|=\|\mathbf{a}_0\|+\epsilon\frac{<\mathbf{a}_0,\mathbf{a}_\epsilon>}{\|\mathbf{a}_0\|}$:

$$\begin{aligned}\|\hat{\mathbf{a}}\|&=\frac{t}{2}\theta_0+\epsilon\frac{\big(\frac{t}{2}\big)^2\theta_0\theta_\epsilon}{\frac{t}{2}\theta_0} &(1.15)\\ &=\frac{t}{2}\theta_0+\epsilon\frac{t}{2}\theta_\epsilon &(1.16)\\ &=\frac{t}{2}(\theta_0+\epsilon\theta_\epsilon)=\frac{t}{2}\hat{\theta} &(1.17)\end{aligned}$$

Finally, by 1.4, 1.17 and $e^{\hat{\mathbf{a}}} = \cos(\|\hat{\mathbf{a}}\|) + \frac{\hat{\mathbf{a}}}{\|\hat{\mathbf{a}}\|}\sin(\|\hat{\mathbf{a}}\|)$,

$$\hat{\mathbf{q}}^t = e^{\hat{\mathbf{a}}} \tag{1.18}$$

$$= \cos\left(\frac{t}{2}\hat{\theta}\right) + \frac{\hat{\mathbf{a}}}{\frac{t}{2}\hat{\theta}}\sin\left(\frac{t}{2}\hat{\theta}\right) \tag{1.19}$$

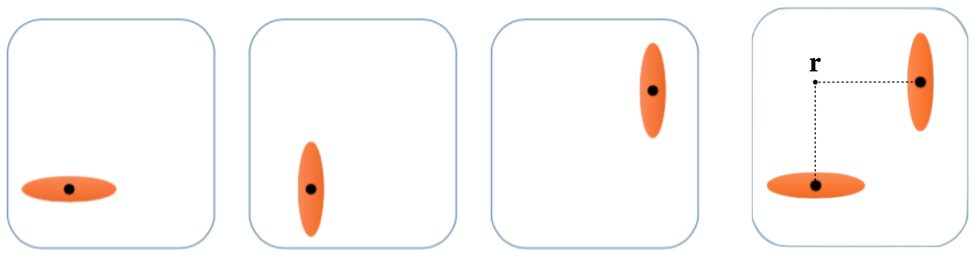
$$= \cos\left(\frac{t}{2}\hat{\theta}\right) + \frac{\cancel{\frac{t}{2}}\cancel{\hat{\theta}}\,\hat{\mathbf{s}}}{\cancel{\frac{t}{2}}\cancel{\hat{\theta}}}\sin\left(\frac{t}{2}\hat{\theta}\right) \tag{1.20}$$

$$= \cos\left(\frac{t}{2}\hat{\theta}\right) + \hat{\mathbf{s}}\sin\left(\frac{t}{2}\hat{\theta}\right) \tag{1.21}$$

The proof has been done.

- Consider rigid transformations in the 2D xy-plane. For these transformations, the rotation is always around the z (or $-z$)-axis, i.e. $\mathbf{s}_0$ is fixed to the z-axis. On the other hand, a dual quaternion encodes translations only along $\mathbf{s}_0$, which are in this case always zero, since we can only translate in the xy-plane. Then, how can a dual quaternion represent a rotation and translation in the xy-plane, such as the one depicted in Figure 1.2a?

Figure 1.1: Dual quaternion (or screw) representation of 2D translation



(a) An object (left) is first rotated around its center of mass (middle) and then translated (right)

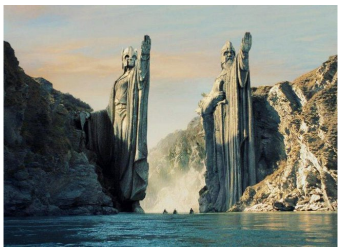
(b) Shifting the screw axis

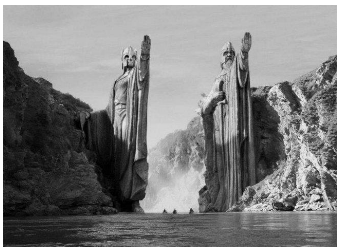
For the 2D case of Figure 1.2a, if the axis of dual quaternion(screw) is shifted to point $\mathbf{r}$ as shown in Figure 1.2b, such transformation can be represented as a screw axis i.e. dual quaternion. In this case, $\mathbf{s}_\epsilon = \mathbf{r} \times \mathbf{s}_0$. [1]

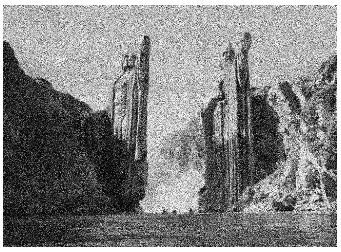
2 Exercise part 2: Variational Methods - Denoising problems

In this section, three different denoising methods(filtering, heat diffusion and variational approach) were implemented and compared.

Figure 2.1: Original images and noised image


(a) Original image

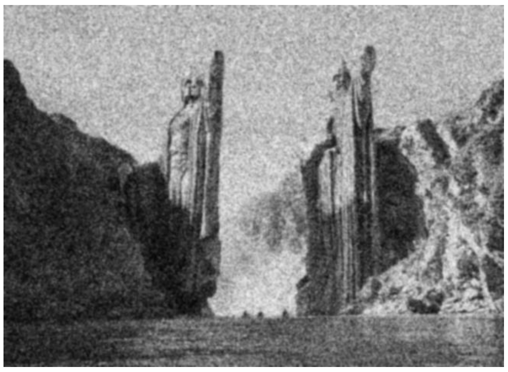

(b) Grayscale image

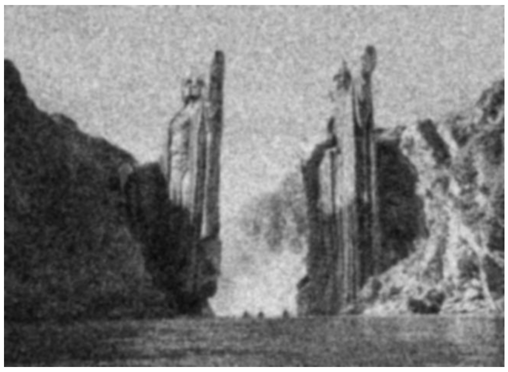

(c) Noised image

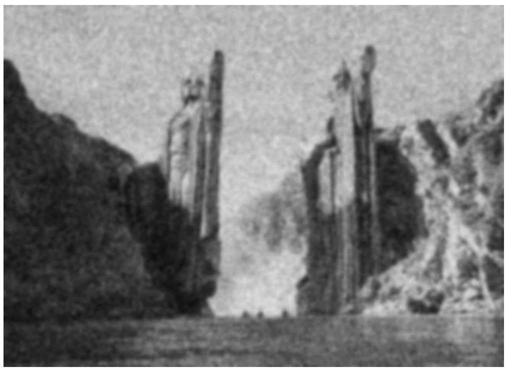
2.1 Task 1: Filtering

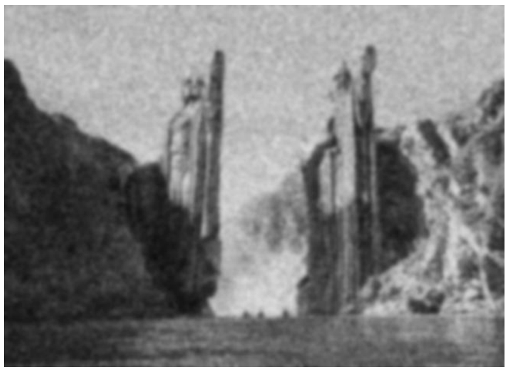
By convolution with Gaussian filter($\sigma = 0.5$), the gaussian noise was denoised.

Figure 2.3: Denoised image after filtering multiple time ($\sigma = 0.5$)


(a) Filtered 8 times


(b) Filtered 16 times


(c) Filtered 24 times


(d) Filtered 32 times

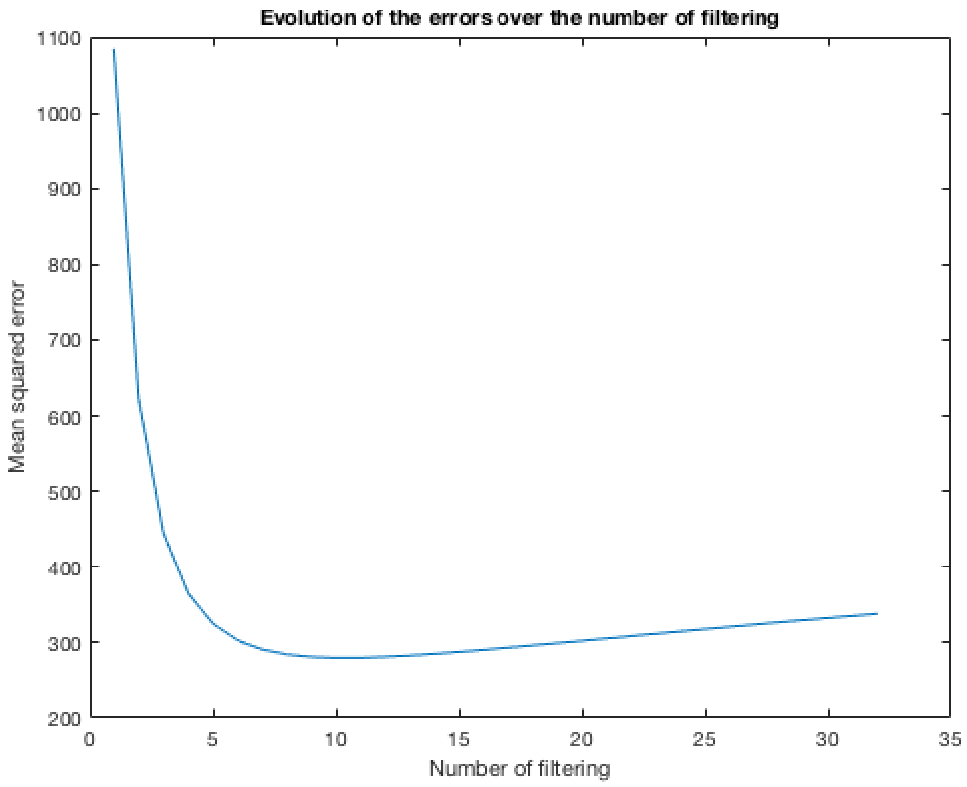
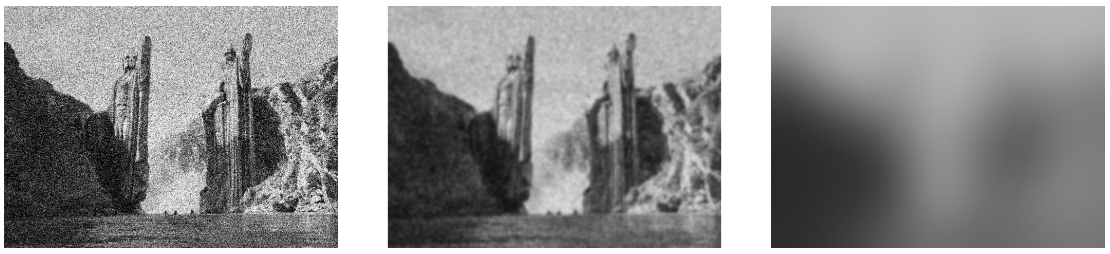


Figure 2.5: Mean square error evolution of Gaussian filtering ($\sigma = 0.5$)

As the iteration goes on, mean square error(MSE) changes as Figure 2.5. For $\sigma = 0.5$, the MSE is the smallest when the number of filtering is 10.

Results with different value of σ are as Figure 2.6. σ determines level of smoothing. If σ is too small or too large, filtering does not show effective denoising.

Figure 2.6: Denoised image with different σ



(a) $\sigma = 0.1$ (32 iter)   (b) $\sigma = 0.5$ (32 iter)   (c) $\sigma = 10$ (32 iter)

2.2 Task 2: Heat diffusion

Let I_t the image at scale t, then heat equation is defined as follows:

$$\frac{\partial I_t}{\partial t} - \Delta I_t = 0 \tag{2.1}$$

where Δ is the discrete 2D laplacian operator. For discretizing and solving 2.1, forward finite differences were used:

$$\frac{I_{t+1} - I_t}{\tau} = \Delta I_t \tag{2.2}$$

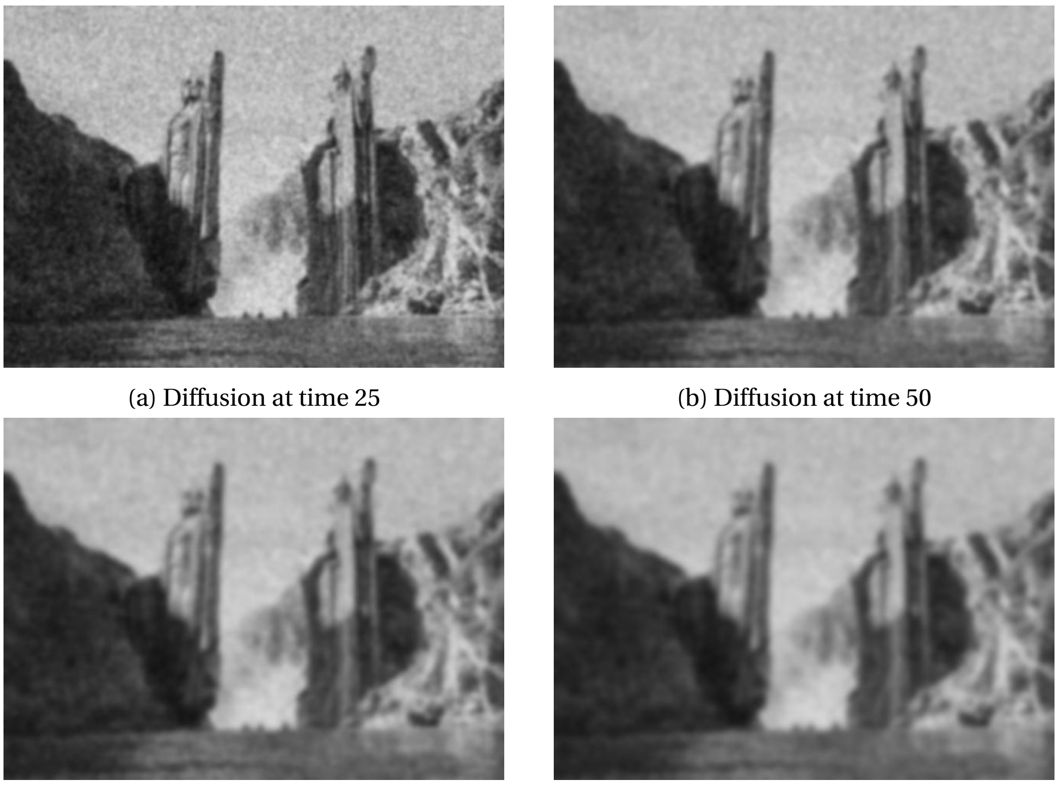
$$I_{t+1} = I_t + \tau \cdot \Delta I_t \tag{2.3}$$

For 2D Laplacian at pixel (x, y), the following was used:

$$\Delta I(x,y) = I(x+1,y) + I(x-1,y) + I(x,y+1) + I(x,y-1) - 4 \cdot I(x,y) \tag{2.4}$$

Figure 2.8 shows the result images with different number of iteration of equation 2.3. Neumann boundary condition was used for boundary padding.

Figure 2.8: Denoised image after diffusion


(a) Diffusion at time 25 (b) Diffusion at time 50
(c) Diffusion at time 75 (d) Diffusion at time 100

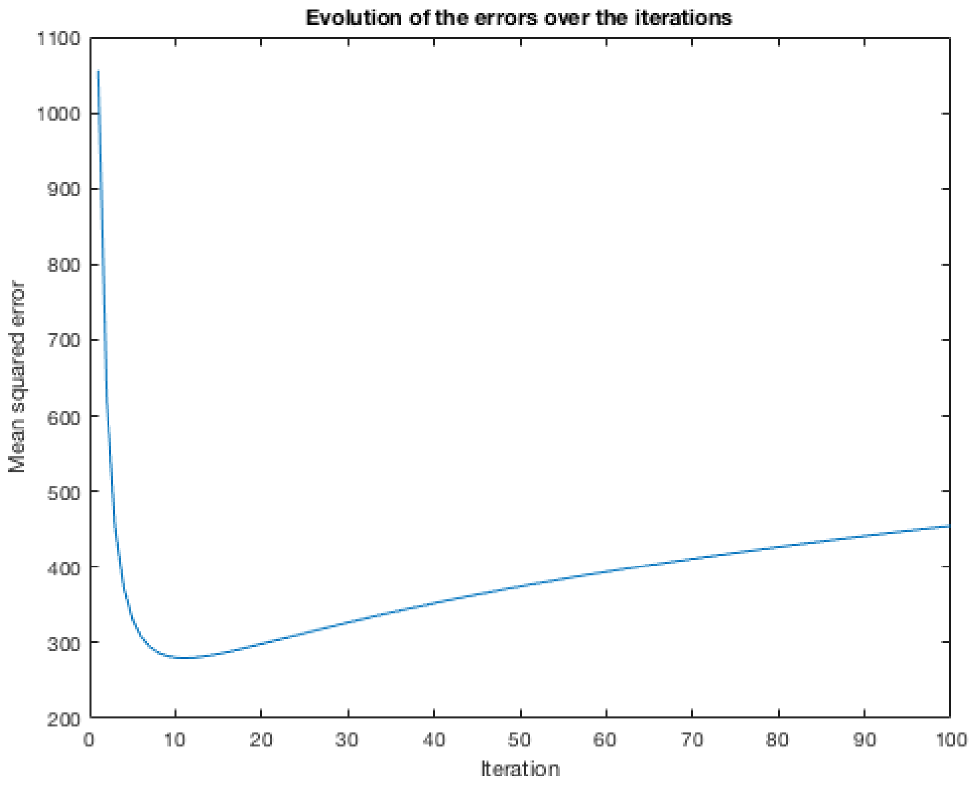
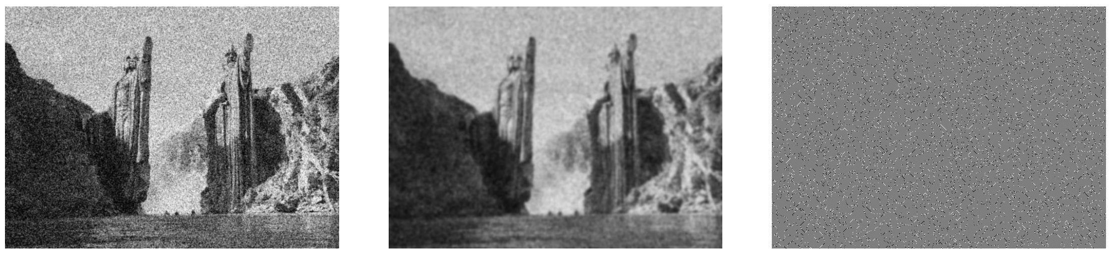


Figure 2.10: Mean square error evolution of heat diffusion ($\tau = 0.1$)

As the iteration goes on, mean square error(MSE) evolves as Figure 2.10. For $\tau = 0.1$, the MSE is the smallest when the number of iteration is 11. (MSE = 279.9)

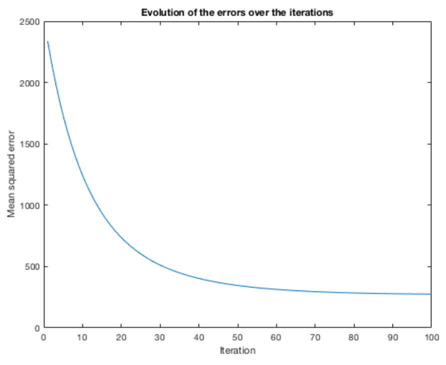
Results with different value of τ are as Figure 2.11. MSE evolution is as 2.13. If τ is too small, it takes longer time to find a optimal solution. In contrast, if τ is too large, the MSE diverges as Figure 2.12c and 2.14b.

Figure 2.11: Denoised image with different τ

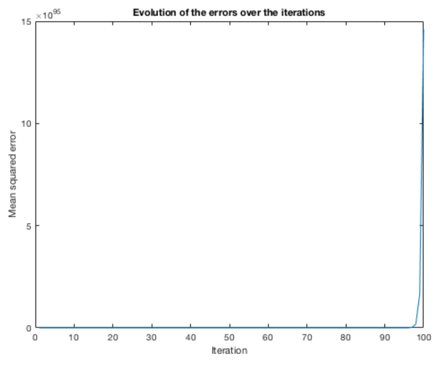


(a) $\tau = 0.01$ (25 iter) (b) $\tau = 0.1$ (25 iter) (c) $\tau = 0.5$ (25 iter)

Figure 2.13: Denoised image with different τ



(a) $\tau = 0.01$



(b) $\tau = 0.5$ (diverged)

2.3 Task 3: Variational approach

2.3.1 Description

Finally, variational approach was implemented. This method considers the image as function of the space of all images.

Derivation of Euler-Lagrange equation The energy function for denoising problem can be defined as follows:

$$E(I) = \int_\Omega \left[\left(I(\mathbf{x}) - I_0(\mathbf{x}) \right)^2 + \lambda \left\| \nabla_{\mathbf{x}} I(\mathbf{x}) \right\|^2 \right] d\mathbf{x} \tag{2.5}$$

where, $\Omega = \mathbb{R}^2$ is the domain of the 2D image, I_0 the noisy image, and λ a regularization parameter. $I, I_0 \in \mathcal{V} = \mathcal{L}^2(\Omega)$.

Let's define the function $L(I, \nabla_{\mathbf{x}} I, \mathbf{x})$ as follows:

$$L(I, \nabla_{\mathbf{x}} I, \mathbf{x}) = \left[\left(I(\mathbf{x}) - I_0(\mathbf{x}) \right)^2 + \lambda \left\| \nabla_{\mathbf{x}} I(\mathbf{x}) \right\|^2 \right] \tag{2.6}$$

Then the Gâteaux derivative is given by

$$\delta E(I; h) = \lim_{\alpha \to 0} \frac{1}{\alpha} \Big(E(I + \alpha h) - E(I) \Big) \tag{2.7}$$

$$= \lim_{\alpha \to 0} \frac{1}{\alpha} \int_\Omega \Big(L(I + \alpha h, \nabla_{\mathbf{x}} I + \alpha \nabla_{\mathbf{x}} h, \mathbf{x}) - L(I, \nabla_{\mathbf{x}} I, \mathbf{x}) \Big) d\mathbf{x} \tag{2.8}$$

apply matrix Taylor expansion:

$$= \lim_{\alpha \to 0} \frac{1}{\alpha} \int_\Omega \Big(\cancel{L(I, \nabla_{\mathbf{x}} I, \mathbf{x})} + \alpha h \cdot \frac{\partial L}{\partial I} + \alpha \nabla_{\mathbf{x}} h \bullet \frac{\partial L}{\partial (\nabla_{\mathbf{x}} I)} + o(\alpha^2) - \cancel{L(I, \nabla_{\mathbf{x}} I, \mathbf{x})} \Big) d\mathbf{x} \tag{2.9}$$

$$= \int_\Omega \Big(h \cdot \frac{\partial L}{\partial I} + \nabla_{\mathbf{x}} h \bullet \frac{\partial L}{\partial (\nabla_{\mathbf{x}} I)} \Big) d\mathbf{x} \tag{2.10}$$

apply integration by parts and $h = 0$ on boundary :

$$= \int_\Omega h \cdot \frac{\partial L}{\partial I}\, d\mathbf{x} + \cancelto{0}{h \cdot \frac{\partial L}{\partial(\nabla_\mathbf{x} I)}\Big|_\Omega} - \int_\Omega h \cdot \nabla_\mathbf{x} \bullet \frac{\partial L}{\partial(\nabla_\mathbf{x} I)}\, d\mathbf{x} \tag{2.11}$$

$$= \int_\Omega h(\mathbf{x}) \cdot \Big(\frac{\partial L}{\partial I} - \nabla_\mathbf{x} \bullet \frac{\partial L}{\partial(\nabla_\mathbf{x} I)}\Big)\, d\mathbf{x} \tag{2.12}$$

Remark following theorem:

Theorem 1. *If $\hat{u}$ is an extremum of a functional $E : \mathcal{V} \to \mathbb{R}$, then*

$$\delta E(\hat{u}, h) = 0 \quad \forall h \in \mathcal{V}.$$

By equation 2.12 and **Theorem 1**,

$$\frac{\partial L}{\partial I} - \nabla_\mathbf{x} \bullet \frac{\partial L}{\partial(\nabla_\mathbf{x} I)} = 0 \tag{2.13}$$

Plug 2.6 into 2.13:

$$\frac{\partial L}{\partial I} = 2\big(I - I_0\big) \tag{2.14}$$

$$\nabla_\mathbf{x} \bullet \frac{\partial L}{\partial(\nabla_\mathbf{x} I)} = 2\lambda \nabla_\mathbf{x} \bullet (\nabla_\mathbf{x} I) \tag{2.15}$$

$$= 2\lambda\, \mathbf{div}(\nabla_\mathbf{x} I) \tag{2.16}$$

Thus,

$$2(I - I_0) - 2\lambda\, \mathbf{div}(\nabla_\mathbf{x} I) = 0 \tag{2.17}$$

$$I_0 = I - \lambda\, \mathbf{div}(\nabla_\mathbf{x} I) \tag{2.18}$$

Vectorization and Linear operation Let I is a 2D image which of height (h) × width (w). As 2D Laplacian is defined as equation 2.4, equation 2.18 can be described as follows:

$$I_0(x, y) = I(x, y) - \lambda\big[I(x+1, y) + I(x-1, y) + I(x, y+1) + I(x, y-1) - 4 \cdot I(x, y)\big] \tag{2.19}$$

In order to make 2.19 as a linear equation, reshape I and I_0 as a 1D: $\bar{I}$ and $\bar{I}_0$:

$$\bar{I}_0(\bar{x}) = \bar{I}(\bar{x}) - \lambda\big[\bar{I}(\bar{x}+h) + \bar{I}(\bar{x}-h) + \bar{I}(\bar{x}+1) + \bar{I}(\bar{x}-1) - 4 \cdot \bar{I}(\bar{x})\big] \tag{2.20}$$

$$= (1 + 4\lambda)\bar{I}(\bar{x}) - \lambda\bar{I}(\bar{x}+h) - \lambda\bar{I}(\bar{x}-h) - \lambda\bar{I}(\bar{x}+1)\lambda\bar{I}(\bar{x}-1) \tag{2.21}$$

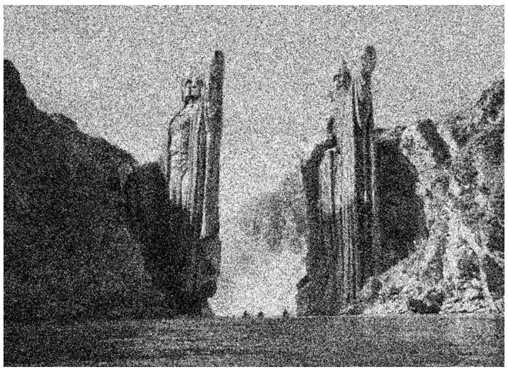
Thus, for matrix A which is defined on the manual, following equation is valid (boundary condition is ignored):

$$\bar{I}_0 = A\bar{I} \tag{2.22}$$

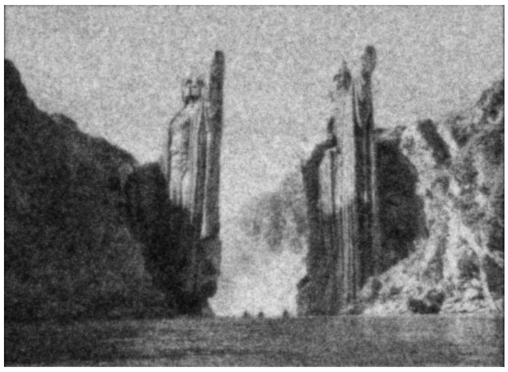
2.3.2 Results

Figure 2.16b is a denoised image by variational method with $\lambda = 2$

Figure 2.15: Denoised image with variational method



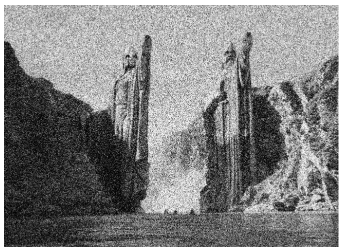
(a) Noisy image



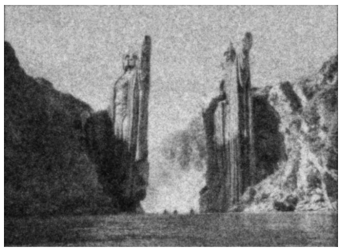
(b) Variational denoising

The results with different values of λ are as Figure 2.17.

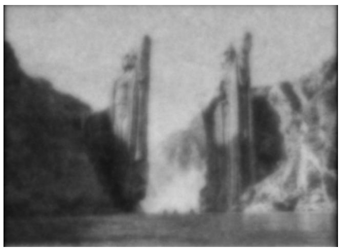
Figure 2.17: Denoised image with different λ



(a) $\lambda = 0.1$ (MSE = 530.9)



(b) $\lambda = 2$ (MSE = 311.5)



(c) $\lambda = 20$ (MSE = 1416.4)

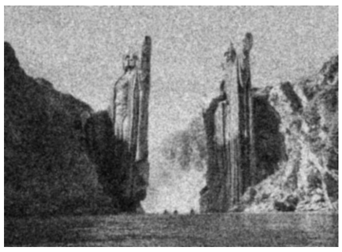
2.4 Task 4: Comparison

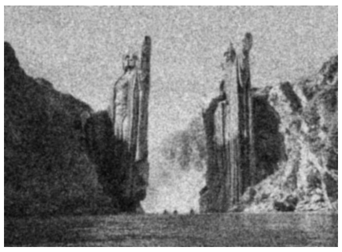
- How can you describe the results? Does any of these methods give better results than the others?

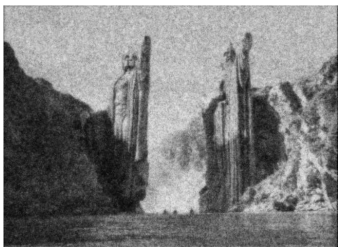
  - For given $\sigma = 0.5$, the minimum MSE result by Gaussian filtering is when the number of iteration is 10. (MSE = 279.2) More iteration leads to blurred image.
  - For given $\tau = 0.1$, 11 iteration follows minimum MSE = 279.9. More iteration leads to blurred image.
  - For given $\lambda = 2$, MSE = 311.5.

  - As Figure 2.19, denoising results with the best number of iterations are very similar. However, for filtering and heat diffusion methods, it is crucial to determine the number of iterations since as iteration goes on, image is getting blurred. Also since σ and τ are hidden parameter, in other words, it's hard to adjust those parameters. Besides, variational method is not iterative method, thus it does not need to determine the number of iteration. Only one for a clear result is a proper value of λ thus easy to choose.

Figure 2.19: The best denoised images by each method


(a) Gaussian filtering $\sigma = 0.5$ (iter = 10, MSE = 279.2)


(b) Heat diffusion $\tau = 0.1$ (iter = 11, MSE = 279.9)


(c) Variational method $\lambda = 2$ (MSE = 311.5)

- What are the benefits and drawbacks of each methods?
  - Gaussian filtering
    * Simple and intuitive way for denoising (+)
    * **Kernel size σ is crucial. Which is a hidden parameter. (-)**
    * **It's hard to define criterion of global optimal solution of denoising. (-)**
    * Boundary condition is not intuitive. (-)
    * **Iterative method i.e. time consuming and hard to determine the number of iterations. (-)**
  - Heat diffusion
    * Boundary condition is more intuitive. Neumann boundary condition is reasonable assumption. (+)
    * **It's hard to define criterion of global optimal solution of denoising. (-)**
    * **Time step τ is crucial. Small value leads to huge amount of time. Even error diverges with large value of τ. Moreover, it is a hidden parameter. (-)**
    * **Iterative method i.e. time consuming and hard to determine the number of iterations. (-)**
  - Variational method
    * **Transparent formulation with global optimal solution criterion. (+)**
    * **No σ or τ. λ is easy to optimize. (+)**

  * **Not iterative method but linear equation. Much faster. (+)**
  * Non-linear diffusivity (e.g. edge preservation.) (+)
  * Big image requires big size of matrix A. (-)
  * Mathematically complicate. (-)

- Can you explain the motivations behind each of the methods?
  - Gaussian filtering

    Using a Gaussian kernel is the simplest and the most intuitive way for denoising, since it works as a low pass filter in frequency domain and noise is mostly high frequency. Thus only low frequency elements remain.
  - Heat diffusion

    Well-known heat diffusion equation can be adapted to the problem. As times goes, noised pixel with high intensity values which correspond to "high temperature area", is denoised by "heat transfer".
  - Variational method

References

[1] L. Kavan, S. Collins, J. Žára, and C. O'Sullivan, "Geometric skinning with approximate dual quaternion blending," *ACM Transactions on Graphics (TOG)*, vol. 27, no. 4, p. 105, 2008.