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: 25.04.2023
Submission deadline: 09.05.2023, 23:59

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 up to 24h will be accepted with a penalty, except for extensions in case of serious illness or emergency. In that case please notify the assistant and provide a relevant medical certificate.

Software. All exercises of this course use Python. See the exercise session slides and this document for hints or specific functions that could be useful for your implementation.

What to hand in. Upload your solution in a .zip file on Moodle. The file must be called "MATHFOUND23-*-firstname-familyname.zip" (replace * with the assignment number). The .zip file MUST contain a single folder called "MATHFOUND23-*-firstname-familyname" with the following data inside:

- A folder named "code" containing your code
- A PDF README / Report file 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 8.3% of your final grade. Your submission will be graded according to the quality of the images produced by your program, the conformance of your program to the expected behaviour of the assignment, and your understanding of the underlying techniques used in the assignment. The submitted code must produce exactly the same images included in your submission (up to randomness).

Goal of this exercise

1. PART 1: Understanding and Utilizing Dual Quaternions

This part of the exercise is devoted to understanding the basic properties of and utilizing dual quaternions when blending rigid transformations, which is required for many applications in computer graphics and vision. Below we will provide all required properties. In case of need, please use Appendix A of the paper *Geometric skinning with approximate dual quaternion blending* (pdf included with the exercise pack) as a reference to understand how rigid transformations are represented with dual quaternions, and their useful properties.

1.1. **Task 1: Thinking about fundamental properties.**

First, please answer the following questions in maximum two sentences for each:

- Question 1: What is the advantage of representing rigid transformations with dual quaternions for blending?
- Question 2: Briefly explain one fundamental disadvantage of using quaternion based shortest path blending for rotations as compared to linear blend skinning (i.e. averaging rotation matrices)? Hint: think about the continuity of both blending methods for 2D rotations.

1.2. **Task 2: Derivations and deeper understanding.**

Below we list all required properties of dual quaternions for the rest of the exercise.

A dual quaternion $\hat{\mathbf{q}}$ can be written in the form $\hat{\mathbf{q}} = \mathbf{q}_0 + \epsilon\mathbf{q}_\epsilon$, where $\mathbf{q}_0$ and $\mathbf{q}_\epsilon$ are quaternions, and ϵ is the dual unit with the property $\epsilon^2 = 0$. The norm of $\hat{\mathbf{q}}$ is then given by $||\hat{\mathbf{q}}|| = ||\mathbf{q}_0|| + \epsilon\frac{<\mathbf{q}_0,\mathbf{q}_\epsilon>}{||\mathbf{q}_0||}$.

Dual quaternions representing rigid transformations can be written in the following form: $\hat{\mathbf{q}} = \cos(\hat{\theta}/2) + \hat{\mathbf{s}}\sin(\hat{\theta}/2)$, where $\hat{\theta} = \theta_0 + \epsilon\theta_\epsilon$ and $\hat{\mathbf{s}} = \mathbf{s}_0 + \epsilon\mathbf{s}_\epsilon$. Here, $\mathbf{s}_0$ is the axis of rotation, θ_0 is the rotation angle, and θ_ϵ is the amount of translation along $\mathbf{s}_0$. Since this is a unit dual quaternion, it can be shown that $< \mathbf{s}_0, \mathbf{s}_\epsilon >= 0$ and $< \mathbf{s}_0, \mathbf{s}_0 >= 1$.

Power of a dual quaternion is defined by $\hat{\mathbf{q}}^t = e^{t\log(\hat{\mathbf{q}})}$, where $e^{\hat{\mathbf{q}}} = \cos(||\hat{\mathbf{q}}||) + \frac{\hat{\mathbf{q}}}{||\hat{\mathbf{q}}||}\sin(||\hat{\mathbf{q}}||)$, and $\log(\cos(\hat{\theta}/2) + \hat{\mathbf{s}}\sin(\hat{\theta}/2)) = \hat{\mathbf{s}}\frac{\hat{\theta}}{2}$.

- Question 3: Utilizing the properties above, for a dual quaternion $\hat{\mathbf{q}} = \cos(\hat{\theta}/2) + \hat{\mathbf{s}}\sin(\hat{\theta}/2)$, prove that $\hat{\mathbf{q}}^t = \cos(t\hat{\theta}/2) + \hat{\mathbf{s}}\sin(t\hat{\theta}/2)$. Hint: you do not need to know the expression for $\cos(\hat{\theta})$ or $\sin(\hat{\theta})$.
- Question 4: Now, 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? Please answer in maximum two sentences without any equations.

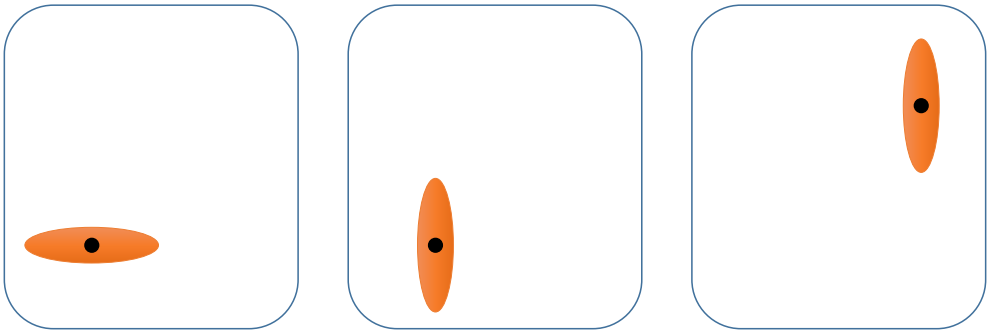

FIGURE 1. An object (left) is first rotated around its center of mass (middle) and then translated (right).

FIGURE 2. (left) Input image (right) Noisy image.

2. PART 2: VARIATIONAL METHODS - DENOISING PROBLEMS

In this section we will study how variational methods can be used in order to solve denoising problems. Application of variational methods come with many benefits both from a practical and a theoretical point of view. The solutions are usually faster and allow a deeper understanding of the mechanism involved during denoising.

The purpose of this exercise is to give you a feeling of these benefits. You will implement and compare three different denoising methods:

- Filtering
- Heat diffusion
- Variational approach

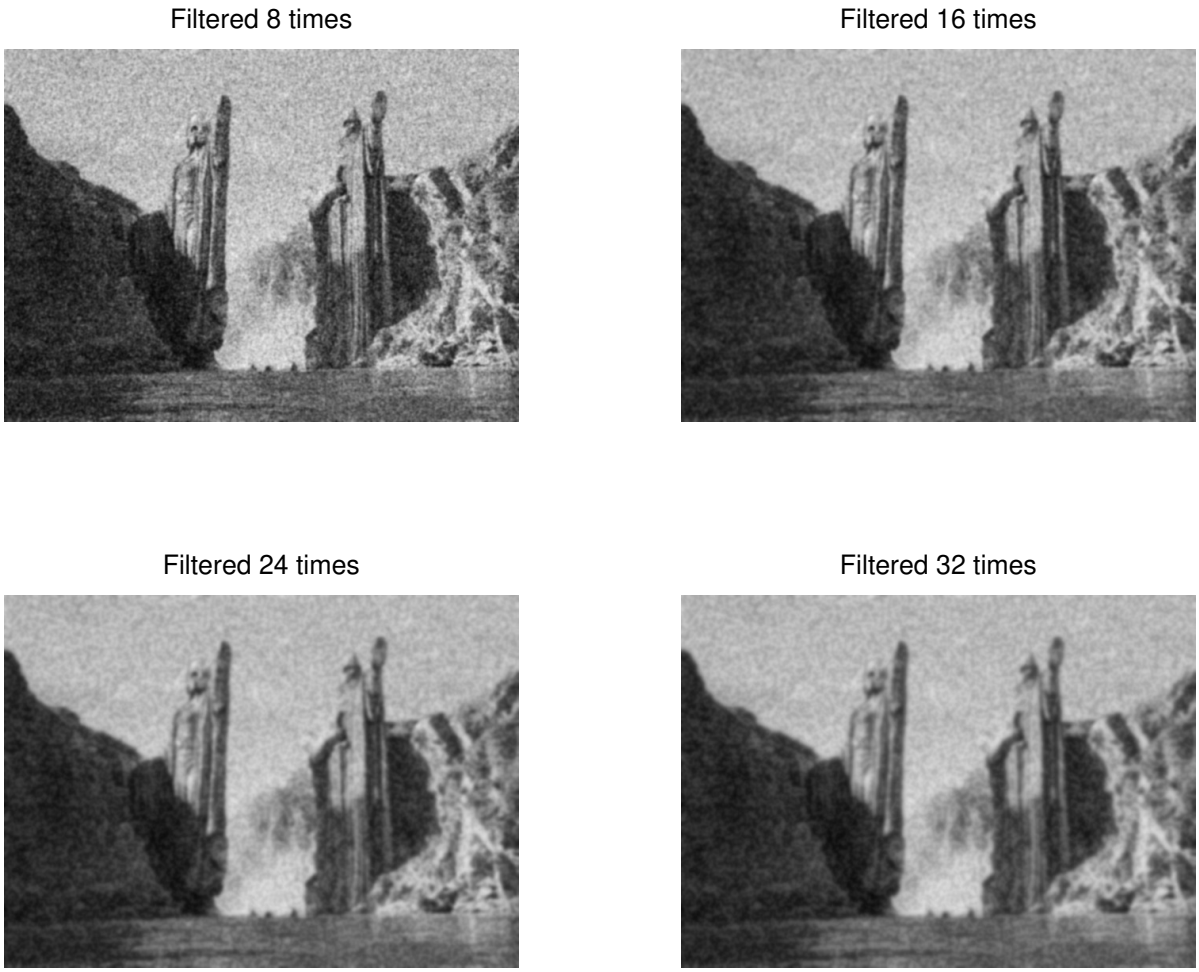

FIGURE 3. Denoised image after filtering multiple time

2.1. **Task 1: Filtering.** Noise is a high frequency signal that is added to the image. Hence one way of removing it is by applying a low-pass filter to it. Usually a gaussian filter is convolved with the image multiple time in order to obtain a reasonable denoised image.

- Create a gaussian filter with $\sigma = 0.5$
- Apply the filter multiple time and show the denoised images at different steps (see figure 3)
- At each step, compute the error and plot its evolution (see figure 4)

Hint.

- The size of your gaussian filter should be $2 \cdot \text{ceil}(3 \cdot \sigma) + 1$ in each direction.

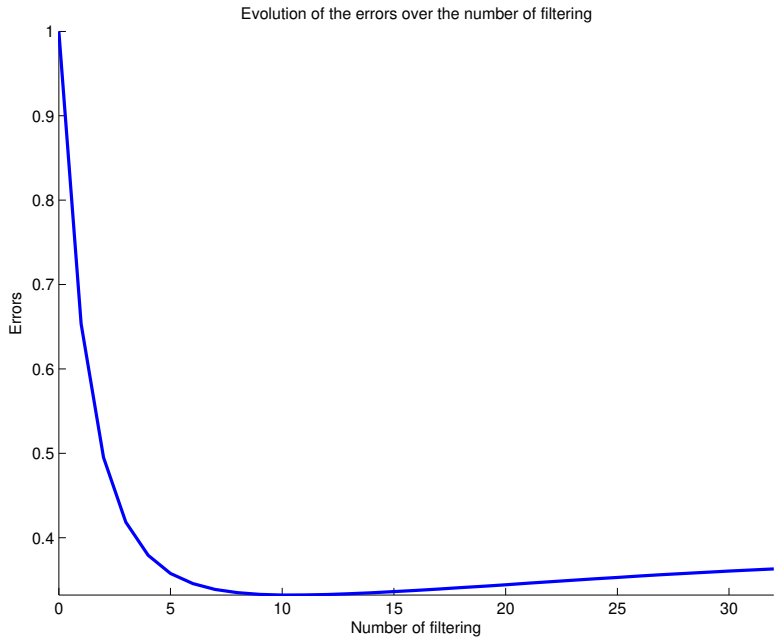

FIGURE 4. Evolution of the error depending on the number of filtering

- You will use a finite filter, therefore you should not use the factor $\frac{1}{2\pi\sigma^2}$ (which is a normalizer for the infinite case), but rather normalize after defining your filter.

Note.

- When doing convolution, you need to pay attention to what is happening at the boundaries. Different boundary conditions can be used, such as for instance zero padding (Dirichlet boundary conditions). In our case, we prefer using Neuman boundary conditions, that is the gradient at the border of the image is set to 0.

2.2. **Task 2: Heat diffusion.** Another approach for denoising an image is to use partial differential equations. The underlying idea is that the image can be considered at different scales. The scale refers here to the level of smoothing. The image is thus represented as a set of smooth images which are identified by a parameter t. Using PDEs allows us to move through this scale space. Let I_t the image at scale t, then we define the following PDE:

$$\frac{\partial I_t}{\partial t} - \Delta I_t = 0 \tag{1}$$

where Δ is the discrete 2D laplacian operator. Equation (1) is the well-known heat equation.

We can use forward finite differences to solve this (1).

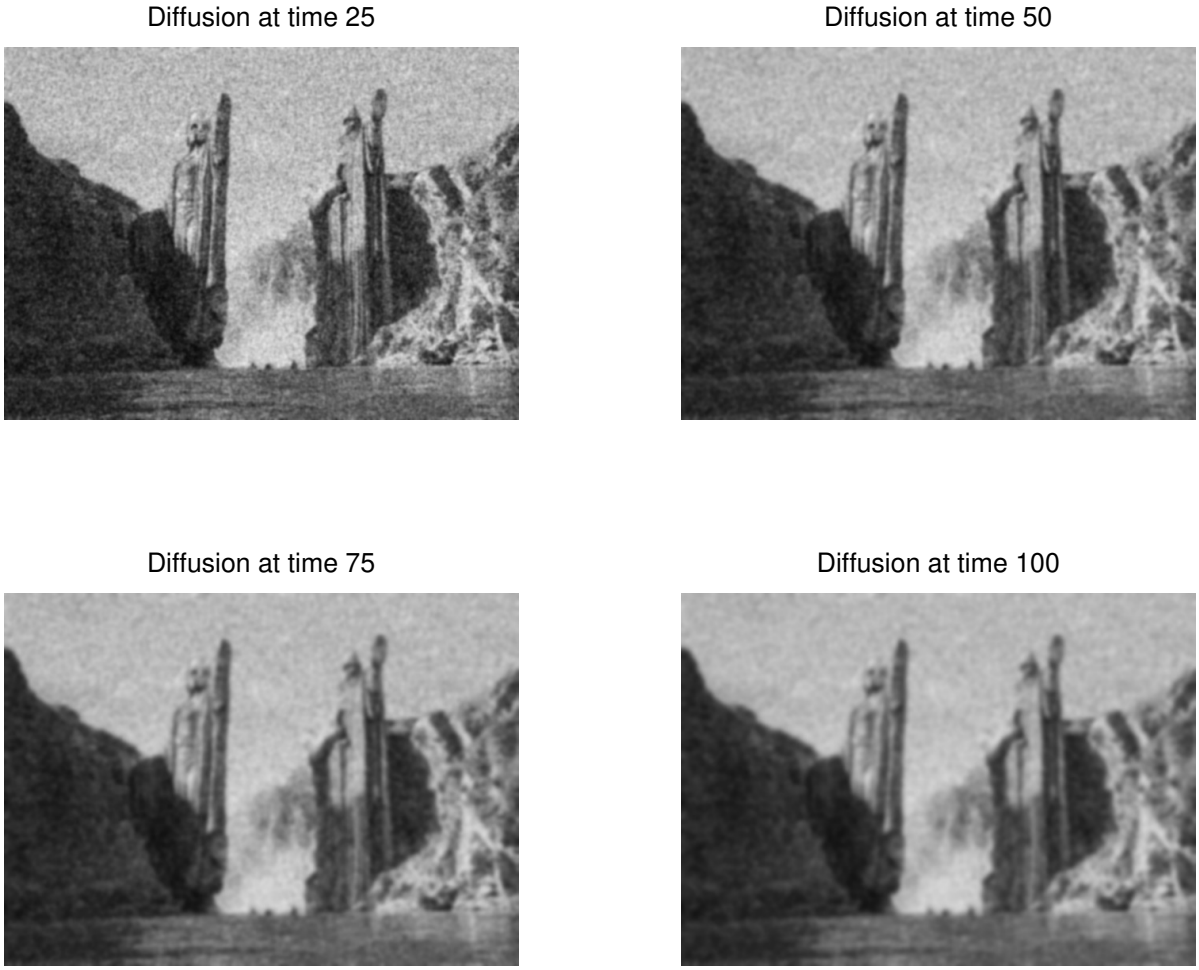

FIGURE 5. Denoised image after diffusion

$$\frac{I_{t+1} - I_t}{\tau} = \Delta I_t \tag{2}$$

$$I_{t+1} = I_t + \tau \cdot \Delta I_t \tag{3}$$

where τ is the time step that should be sufficiently small. As you can see it is an iterative process.

- Implement the heat diffusion for denoising
- Show the evolution of denoising over diffusion by printing denoised images at different steps (see figure 5)
- Show the evolution of the errors over iterations (see figure 6)

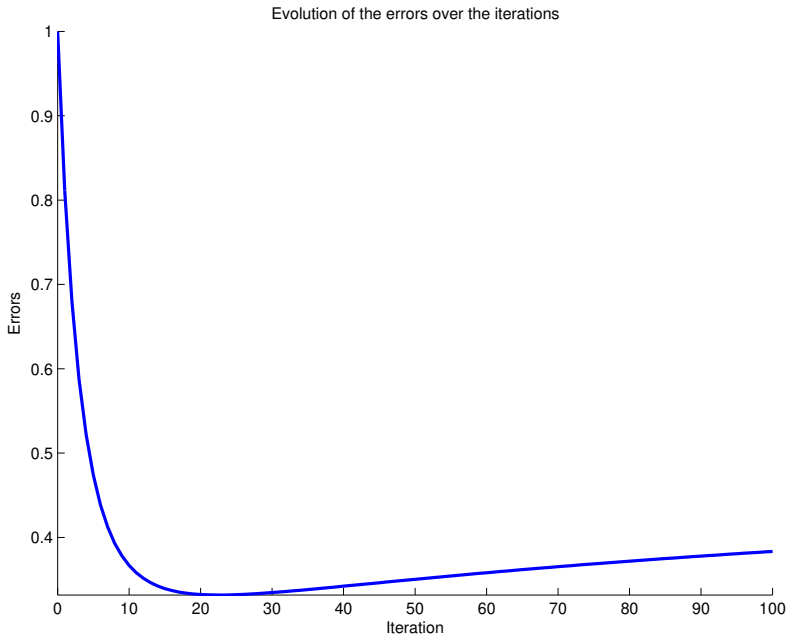

FIGURE 6. Evolution of the error over iteration

Hint.

- The 2D Laplacian at pixel (x, y):

$$\Delta I(x,y) = I(x+1,y) + I(x-1,y) + I(x,y+1) + I(x,y-1) - 4 \cdot I(x,y)$$

- As in the previous task, you must pay attention to the behaviour of the boundaries, and use Neumann boundary conditions (gradient of the image that borders is 0).

2.3. **Task 3: Variational approach.** The last method we will explore is to consider the image as function of the space of all images. We want to find a smooth approximation of the function. To do so we will define an energy functional that we will minimize.

$$E(I) = \int_{\Omega} \left[(I(\mathbf{x}) - I_0(\mathbf{x}))^2 + \lambda \parallel \nabla I(\mathbf{x}) \parallel^2 \right] d\mathbf{x} \tag{4}$$

where Ω is the domain of the image, I_0 the noisy image, and λ a regularization parameter. After calculus of variations we obtain the following equation:

$$I_0 = I_u - \lambda \cdot \mathbf{div}(\nabla I_u) \tag{5}$$

which can be rewritten as:

$$\bar{I}_0 = A\bar{I}_u \tag{6}$$

where $\bar{I}$ is the vectorization of I, I is of size $H \times W$. For simplicity, we will use this case the Dirichlet boundary conditions which gives the matrix A the following form

$$\begin{pmatrix}
a & b & 0 & \cdots & 0 & b & 0 & \cdots & & & & & & \\
b & a & b & 0 & \cdots & 0 & b & 0 & \cdots & & & & & \\
0 & b & \ddots & \ddots & \ddots & & \ddots & \ddots & \ddots & & & & & \\
\vdots & \ddots & \ddots & \ddots & \ddots & \ddots & & \ddots & & & & & & \\
0 & \cdots & 0 & b & a & b & 0 & \cdots & 0 & b & 0 & & & \\
b & 0 & \cdots & 0 & b & a & b & 0 & \cdots & 0 & b & 0 & & \\
0 & b & \ddots & & \ddots & \ddots & \ddots & \ddots & \ddots & & \ddots & \ddots & \ddots & \\
\vdots & \ddots & \ddots & & & \ddots & \ddots & \ddots & \ddots & \ddots & & \ddots & \ddots & \ddots \\
 & \ddots & 0 & b & 0 & \cdots & 0 & b & a & b & 0 & \cdots & 0 & b \\
 & & & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & & \ddots \\
 & & & & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \\
 & & & & & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\
 & & & & & & 0 & b & 0 & \cdots & 0 & b & a & b \\
 & & & & & & & 0 & b & 0 & \cdots & 0 & b & a
\end{pmatrix}$$

with

$$a = 1 + 4\lambda$$
$$b = -\lambda$$

Your tasks will be the following:

- Derive the energy (4) to obtain the Euler-Lagrange equation (5)
- Solve the variational problem for $\lambda = 2$
- Show the denoised image that you obtain (see figure 7)

2.4. **Task 4: Comparison / Questions.**

- When running the gaussian convolution over and over, does the solution converge? If so, what would be the steady state? Does it depend on the boundary conditions (Neumann vs Dirichlet)?
- If you apply heat diffusion infinitely, does it converge to a steady state? If so, what would be the steady state? Does it depend on the boundary conditions (Neumann vs Dirichlet)?
- What is the Euler Lagrange equation of an energy? What is its purpose?

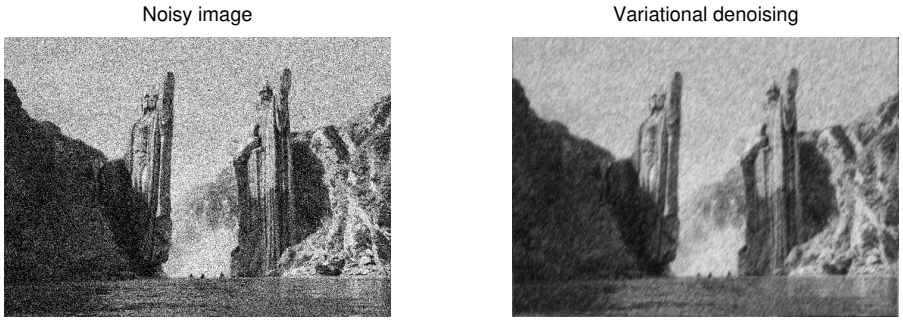

FIGURE 7. Denoised image with variational method

- Is the solution of the Euler-Lagrange equation globally optimal? What parameter can you modify to change the level of smoothing? Explain in your own words how it works.
- How can you describe the results? Does any of these methods give better results than the others? Explain briefly your answer.
- What are the benefits and drawbacks of each methods?
- Can you explain the motivations / intuitions behind each of the methods?