Assignment 1

Simon Jenni
09-116-005

Superresolution

1. **Problem.** The goal of superresolution is to recover high resolution images of given low resolution input images. This problem can be modelled mathematically by assuming the following image creation model:

$$g(x,y) = (u * k)(\alpha x, \alpha y). \tag{1}$$

Here, g is the low resolution input image that results from down-sampling a higher resolution image u with an averaging filter k and down-sampling factor α. The task of superresolution therefore is to find u given g, k and α.

As we deal with images in the discrete, equation 1 can be rewritten as

$$g = Du. \tag{2}$$

In this case, g and u are column vectors (column-major ordering of the images) and D is a down-sampling matrix. The solution u can now be found by minimising

$$E(u) = \frac{\lambda}{2}||Du - g||^2 + ||\nabla u|| \tag{3}$$

with respect to u. Here, $||\nabla u||$ is equal to a Total Variation (TV) regularisation term which ensures a certain smoothness in the output image u and the parameter λ controls the trade-off between minimising the error and regularisation terms. The image we are looking for is therefore given by

$$\tilde{u} = \arg\min_u E(u) \tag{4}$$

2. **Motivations.** There are many applications of superresolution we can think of. Any situation where the highest possible resolution of a screen is

higher than the resolution of some image is a possible use-case. While being more computationally expensive than a simple interpolation scheme, superresolution results in a higher perceived quality of the up-scaled images and allows humans to better interpret these. Superresolution could also be used to save space while storing or transmitting images. Concrete use-cases are mobile-phone images used as wallpaper, the enlargement of a sub window of an image, retrieval of mug shots from surveillance cameras, and so on.

3. **Derivation of gradient.** To numerically solve for the best image u in equation 4, the energy given by equation 3 is discretised. Using forward finite differences for the approximation leads to:

$$E(u) = \frac{\lambda}{2} \sum_{i,j} ((Du)(i,j) - g(i,j))^2 + \sum_{i,j} \tau(i,j) \tag{5}$$

where

$$\tau(i,j) = \sqrt{(u(i+1,j) - u(i,j))^2 + (u(i,j+1) - u(i,j))^2}. \tag{6}$$

To minimise equation 5 we use a gradient-descent approach with the update rule

$$u := u - \alpha \nabla_u E. \tag{7}$$

This of course requires the computation of the gradient $\nabla_u E$ of equation 5 with respect to u:

$$\nabla_u E = \lambda D^T (Du - g) + \frac{\partial}{\partial u} ||\nabla u|| \tag{8}$$

where the entries of $\frac{\partial}{\partial u}||\nabla u||$ in turn are computed by

$$\frac{\partial ||\nabla u||}{\partial u(i,j)} = \frac{\partial \tau(i,j)}{\partial u(i,j)} + \frac{\partial \tau(i-1,j)}{\partial u(i,j)} + \frac{\partial \tau(i,j-1)}{\partial u(i,j)} \tag{9}$$

with the partial derivatives of τ w.r.t u given by:

$$\frac{\partial \tau(i,j)}{\partial u(i,j)} = \frac{2u(i,j) - u(i+1,j) - u(i,j+1)}{\tau(i,j)} \tag{10}$$

$$\frac{\partial \tau(i-1,j)}{\partial u(i,j)} = \frac{u(i,j) - u(i-1,j)}{\tau(i-1,j)} \tag{11}$$

$$\frac{\partial \tau(i,j-1)}{\partial u(i,j)} = \frac{u(i,j) - u(i,j-1)}{\tau(i,j-1)} \tag{12}$$

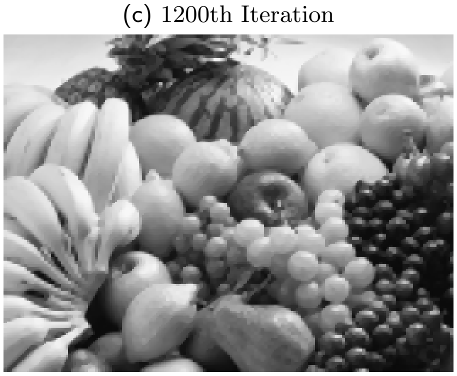
4. **Implement gradient descent for superresolution.** I implemented a gradient descent algorithm to solve for u. The computation of the gradient requires to make some boundary assumptions for u. The assumptions made here are that u is mirrored along its boundaries. The updating-factor α in equation 7 is determined via backtracking line search. Figure 1 shows results at different stages of the minimisation.

(a) 1st Iteration

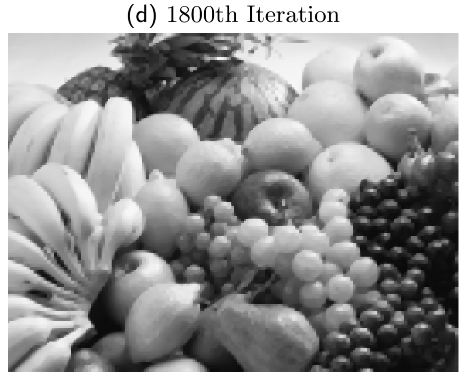
(b) 600th Iteration

(c) 1200th Iteration

(d) 1800th Iteration

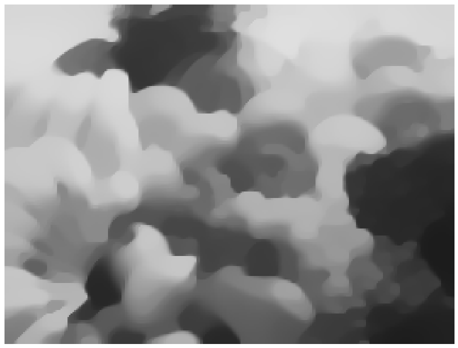


(e) 2400th Iteration



(f) Final Iteration (3000th)

Fig. 1: This figure shows an image at several different stages of the gradient descent algorithm.



(a) $\lambda = 10$ (low)

(b) $\lambda = 1'000'000$ (high)

(c) $\lambda = 10'000$ (reasonable)

Fig. 2: This figure shows results for different values of the parameter λ.

5. **The influence of λ.** The parameter λ in equation 3 controls the trade-off between minimising the error and minimising the regularisation term $||\nabla u||$. Figure 2 shows results for different values of λ. Setting it low places more weight on minimising $||\nabla u||$, therefore making u smoother. Setting λ high on the other hand places more weight on minimising the error between the low-res input image and the down-sampled prediction u, therefore u will be more similar to an up-scaled version of the low-res input image (it will look more pixelated).

6. **Find optimal λ.**

   As we know the optimal value for u, we can look for the value of λ which minimises the sum of squared distances (SSD)

$$SSD(u) = \sum_{i,j} (\tilde{u}(i,j) - u(i,j)) \tag{13}$$

   between the ground truth u and the solution $\tilde{u}$ we computed. Figure 3 shows the SSD vs. λ graph. We observe that very low values for λ result

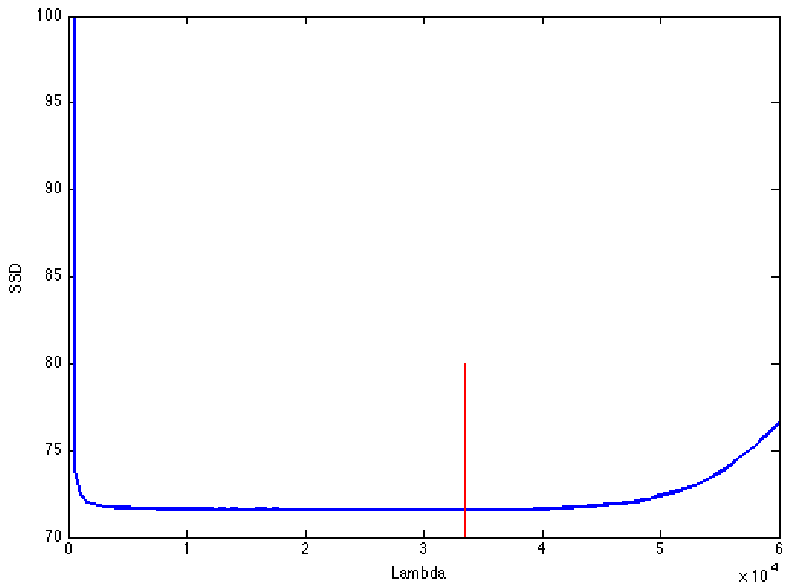


Fig. 3: This figure shows the effect of λ on the SSD between the ground-truth u and our solution $\tilde{u}$. The red vertical line indicates the optimal value for λ.

in a very high SSD. For λ between 5'000 and 40'000, the SSD stays nearly constant and increases again for $\lambda > 40'000$. The graph draws a convex function with a minimum at approximately $\lambda = 33'500$.