Lecture 7: Application to Image Processing

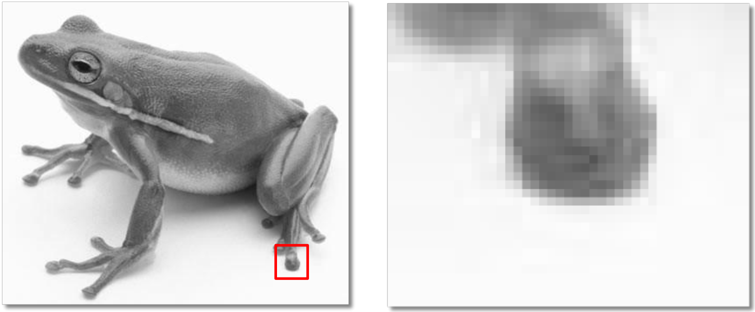

Figure 1: Digital image $u_{i,j}$ is a discretised and quantised version of "real" scene $u(x, y)$

A grey-scale image is simply a two dimensional $N \times M$ array of pixel values $u_{i,j}$, where $i \in 1 \ldots L_x$ and $j \in 1 \ldots L_y$. However, we can also consider that $u_{i,j}$ are samples of some continuous function $u(x, y)$ defined on the domain $(x, y) \in \Omega \subset \mathbb{R}^+$, where $\mathbb{R}^+$ is the set of positive real numbers. For the sake of simplicity, we set the domain so that $\Omega = \{x, y \in \mathbb{R}^+ : x < L_x \wedge y < L_y\}$, so that the distances between neighbouring pixels will be equal to 1.

In this case, an image consists of samples from this continuous function

$$u_{ij} = u(i, j)$$

We can consider a “real” image to consist of both a source term u, which we are trying to recover, and an additive noise term n

$$u_{i,j} = u(i, j) + n(i, j)$$

Sobol Edge Filter

How could you find edges in an image. . . .?

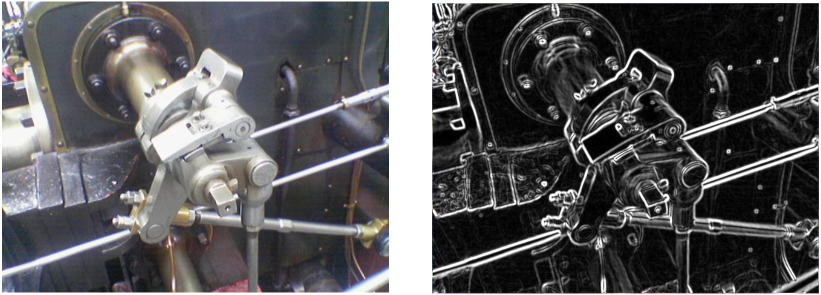

Figure 2: Sobol Edge Detect

The gradient of u provides edge information. When magnitude of $\frac{\partial u}{\partial x}$ is high, likely we are at an edge

$$\left(\frac{\partial u}{\partial x}\right)_i \approx D_c^x u_i = u_{i+1} - u_{i-1}$$

Note we are defining an operator D_c^x that applies the central finite difference formula to u_i

For image processing, u_i values are normally corrupted by noise, which can affect the gradient. We can pre-smooth by triangle filter, e.g.

$$\left(\frac{\partial u}{\partial x}\right)_{i,j} \approx D_c^x (u_{i,j+1} + 2u_{i,j} + u_{i,j-1})$$

We can extend this to a 2D image by taking the magnitude of the (two-dimensional) gradient

$$|\nabla u|_{i,j} \approx \sqrt{\left(\frac{\partial u}{\partial x}\right)^2_{i,j} + \left(\frac{\partial u}{\partial y}\right)^2_{i,j}}$$

[lec07_sobol.m]. Note this shows yet another method to implement the discretisation in MATLAB using index vectors.

Gaussian Blur Noise Reduction

We can reduce the amount of noise in an image by applying the time-dependent heat equation. Noise has a higher curvature than the image, so will be eliminated first

$$u_t = \nabla^2 u = u_{xx} + u_{yy}$$

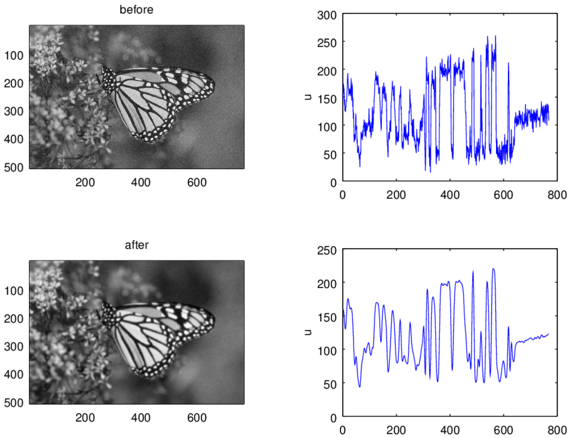

Figure 3: (left) butterfly image, top shows noisy image, bottom shows image after guassian blur (right) plot showing image values through a slice near the centre of the 2D image, top shows values from noisy image,

The time derivative can be approximated using forward differences:

$$u_t(x_i, y_j, t_n) \approx \frac{u_{i,j}^{n+1} - u_{i,j}^{n}}{\Delta t}$$

where $t_n = \Delta t n$ for the set of positive integers $n = 0, \ldots, N$, $x_i = i$ for the set of positive integers $i = 1, \ldots, L$ and $y_j = j$ for the set of positive integers $j = 1, \ldots, L$.

We approximate u_{xx} and u_{yy} using a central difference

$$u_{xx}(x_i, y_j, t_n) \approx \frac{1}{\Delta x^2}(u_{i+1,j}^{n} - 2u_{i,j}^{n} + u_{i-1,j}^{n}) = u_{i+1,j}^{n} - 2u_{i,j}^{n} + u_{i-1,j}^{n}$$

$$u_{yy}(x_i, y_j, t_n) \approx \frac{1}{\Delta y^2}(u_{i,j+1}^{n} - 2u_{i,j}^{n} + u_{i,j-1}^{n}) = u_{i,j+1}^{n} - 2u_{i,j}^{n} + u_{i,j-1}^{n}$$

Putting these into the heat equation gives

$$\begin{aligned}
\frac{u_{i,j}^{n+1} - u_{i,j}^{n}}{\Delta t} &= D(u_{i+1,j}^{n} + u_{i-1,j}^{n} + u_{i,j+1}^{n} + u_{i,j-1}^{n} - 4u_{i,j}^{n}) \\
u_{i}^{n+1} &= u_{i}^{n} + \Delta t D(u_{i+1,j}^{n} + u_{i-1,j}^{n} + u_{i,j+1}^{n} + u_{i,j-1}^{n} - 4u_{i,j}^{n}) \\
u_{i}^{n+1} &= \Delta t D(u_{i+1,j}^{n} + u_{i-1,j}^{n} + u_{i,j+1}^{n} + u_{i,j-1}^{n}) + (1 - 4\Delta t D)u_{i,j}^{n}
\end{aligned}$$

[lec07_diffusion.m]

Spatially Varying Diffusion

The heat equation $u_t = K\nabla u$ involves a diffusion constant K. We could replace that constant with a function of space

$$u_t = \nabla \cdot (K(\mathbf{x})\nabla u). \tag{1}$$

In one dimension, this would be

$$u_t = (K(x)u_x)_x. \tag{2}$$

A common approach to discretizing this is to use a *forward difference* $D_+^x \approx u_x$ on the u_x term

$$u_x \approx D_+^x u = \frac{u_{i+1} - u_i}{\Delta x},$$

then evaluate $K(x)$ at the midpoint: $K(x_{i+\frac{1}{2}})$, so that

$$K(x)u_x \approx K(x_{i+\frac{1}{2}})\frac{u_{i+1} - u_i}{\Delta x},$$

Then, differentiate the result approximately with a *backwards difference* D_-^x.

$$u_x \approx D_-^x u = \frac{u_i - u_{i-1}}{\Delta x},$$

so that

$$\begin{aligned}(K(x)u_x)_x \approx D_-^x(K(x_{i+\frac{1}{2}})D_+^x u) &= \frac{K(x_{i+\frac{1}{2}})\frac{u_{i+1}-u_i}{\Delta x} - K(x_{i-\frac{1}{2}})\frac{u_i - u_{i-1}}{\Delta x}}{\Delta x} \\ &\approx \frac{K(x_{i+\frac{1}{2}})u_{i+1} - (K(x_{i+\frac{1}{2}}) + K(x_{i-\frac{1}{2}}))u_i + K(x_{i-\frac{1}{2}})u_{i-1}}{\Delta x^2}\end{aligned}$$

Naturally, for constant diffusion $K(x) = K$ this reduces to

$$(K(x)u_x)_x \approx K\frac{u_{i+1} - 2u_i + u_{i-1}}{\Delta x^2}$$

2D

For two dimensions, the spatially varying heat equation is

$$u_t = (K(x,y)u_x)_x + (K(x,y)u_y)_y.$$

and this can be discretized in a similar fashion to give

$$(K(x,y)u_x)_x \approx \frac{K(x_{i+\frac{1}{2}},y_j)u_{i+1,j} - (K(x_{i+\frac{1}{2}},y_j) + K(x_{i-\frac{1}{2}},y_j))u_{i,j} + K(x_{i-\frac{1}{2}},y_j)u_{i-1,j}}{\Delta x^2}$$

$$(K(x,y)u_y)_y \approx \frac{K(x_i,y_{j+\frac{1}{2}})u_{i,j+1} - (K(x_i,y_{j+\frac{1}{2}}) + K(x_i,y_{j-\frac{1}{2}}))u_{i,j} + K(x_i,y_{j-\frac{1}{2}})u_{i,j-1}}{\Delta y^2}$$

$$u_t \approx \frac{u^{n+1} - u^n}{\Delta t}.$$

If only the nodal values for $K(x,y)$ are known, then a reasonable approximation is to use the average of two neighbouring grid points

$$K(x_i, y_{j+\frac{1}{2}}) = \frac{1}{2}(K_{i,j+1} + K_{i,j})$$
$$K(x_{i+\frac{1}{2}}, y_j) = \frac{1}{2}(K_{i+1,j} + K_{i,j})$$

Which results in

$$(K(x,y)u_x)_x \approx \frac{(K_{i+1,j} + K_{i,j})u_{i+1,j} - ((K_{i+1,j} + 2K_{i,j} + K_{i-1,j}))u_{i,j} + (K_{i-1,j} + K_{i,j})u_{i-1,j}}{2\Delta x^2}$$
$$(K(x,y)u_y)_y \approx \frac{(K_{i,j+1} + K_{i,j})u_{i,j+1} - ((K_{i,j+1} + 2K_{i,j} + K_{i,j-1}))u_{i,j} + (K_{i,j-1} + K_{i,j})u_{i,j-1}}{2\Delta y^2}$$
$$u_t \approx \frac{u^{n+1} - u^n}{\Delta t}.$$

For $\Delta x = \Delta y = 1$ this simplifies to

$$
\begin{aligned}
u_{i,j}^{n+1} = & u_{i,j}^{n} + \frac{1}{2}\Delta t(\\
& (K_{i+1,j} + K_{i,j})u_{i+1,j} + (K_{i-1,j} + K_{i,j})u_{i-1,j} \\
& (K_{i,j+1} + K_{i,j})u_{i,j+1} + (K_{i,j-1} + K_{i,j})u_{i,j-1} \\
& -(K_{i+1,j} + K_{i-1,j} + K_{i,j+1} + K_{i,j-1} + 4K_{i,j})u_{i,j})
\end{aligned}
$$

Spatially varying diffusion for noise reduction

What shall we use for $K(x, y)$? For noise reduction in images, we may want to preserve image features, or edges. We know that edges are associated with areas of high $|\nabla u|$. We can therefore set $K(x, y) = g(|\nabla u|)$, where g is a given function. Perona & Malik proposed using $g(s) = 1/(1 + \frac{s^2}{\lambda^2})$, why? What is the role of λ?

Recall that,

$$|\nabla u|_{i,j} \approx \sqrt{\left(\frac{\partial u}{\partial x}\right)_{i,j}^2 + \left(\frac{\partial u}{\partial y}\right)_{i,j}^2}$$

therefore we can set

$$\begin{aligned} s_{i,j}^2 &= (D_c^x u_{i,j}^n)^2 + (D_c^y u_{i,j}^n)^2 \\ &= (u_{i+1,j}^n - u_{i-1,j}^n)^2 + (u_{i,j+1}^n - u_{i,j-1}^n)^2 \end{aligned}$$

and finally,

$$K_{i,j} = \frac{1}{1 + \frac{s_{i,j}^2}{\lambda^2}}$$

[lec07_varying_diffusion.m]