Artificial Vision Report - Lab 1

Olivier Jais-Nielsen

March 26, 2009

0.1 C++ functions

0.1.1 Gaussian distribution

The goal of the `gaussianDist` fucntion is to calculate the value of the gaussian distribution$g$ with a zero mean and a given standard deviation σ, according to the following formula : $g(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{x^2}{2\sigma^2}\right)$. Cf. *Algorithm 0.1*

Algorithm 0.1 gaussianDist

```
float gaussianDist(float r, float deviation)
{
        return (exp(-(r*r)/(2*deviation*deviation)))/(
            deviation*sqrt(2*M_PI));
}
```

The arguments are the considered point `r` and the standard deviation `deviation`.

0.1.2 Gaussian mask

The function `gaussianKernel` provides a gaussian mask to be convolved with an image. It takes as input the radius of the mask and the standard deviation of the gaussian distribution it represents. Cf. *Algorithm 0.2*

0.1.3 Main program

The main program gets the image file name as the first console argument (after the name of the program itself). Then it lets the user decide wether it should add noise to the input image. If it is so, it asks for the noise power and uses the method `noise` to apply noise to the image. Then it asks successively for the kernel radius, the standard deviation and wether it should save the modified pictures. It creates a gaussian mask with `gaussianKernel` and covolve it with the image with `get_convolve` ; it doesn't use `convolve` in order to display with two `CImgDisplay` instances the original (or noisy) image next to the blurred image. Finally, if the user has chosen so, it saves the modified images with `save` after having normalized the pixel values with `normalize` so that they can be coded on 8-bit integers. Cf. *figure 0.1*

0.2 Studying the effect of Gaussian filtering

0.2.1 Simple Gaussian filtering

Gaussian filtering blurs the input image : it removes sharp edges (high frequencies). As shown in *figure 0.2*, the bigger the standard deviation is, the more it blurs.

Algorithm 0.2 gaussianKernel

```
CImg<float> gaussianKernel(int radius, float deviation)
{
        CImg<float> mask(2*radius+1, 2*radius+1);
        int i, j;
        float r;
        float I;
        for (i = -radius; i <= radius; i++)
        {
                for (j = -radius; j <= radius; j++)
                {
                        r = i*i + j*j;
                        r = sqrt(r);
                        I = gaussianDist(r, deviation);
                        mask(i + radius, j + radius) = I;
                }
        }
        mask /= mask.norm(1);
        return mask;
}
```

The arguments are the radius `radius` and the standard deviation `deviation`. Each pixel of the mask has the value of the gaussian distribution taken at the distance of the pixel to the center of the mask calculated by `gaussianDist`. Eventually, the mask is normalized so that the sum of its coeficients is equal to 1.

However, if the standard deviation gets too big in comparision to the kernel radius, square artifacts begins to appear (cf. *figure 0.*3).

0.2.2 Gaussian filtering for denoising

As noise is a high frequency term, gaussian filtering logically removes it (cf. *figure 0.4*).

However, if there is too much noise, some higher frequency noise remain after gaussian filtering (cf. *figure 0.*5).

Figure 1: Main program

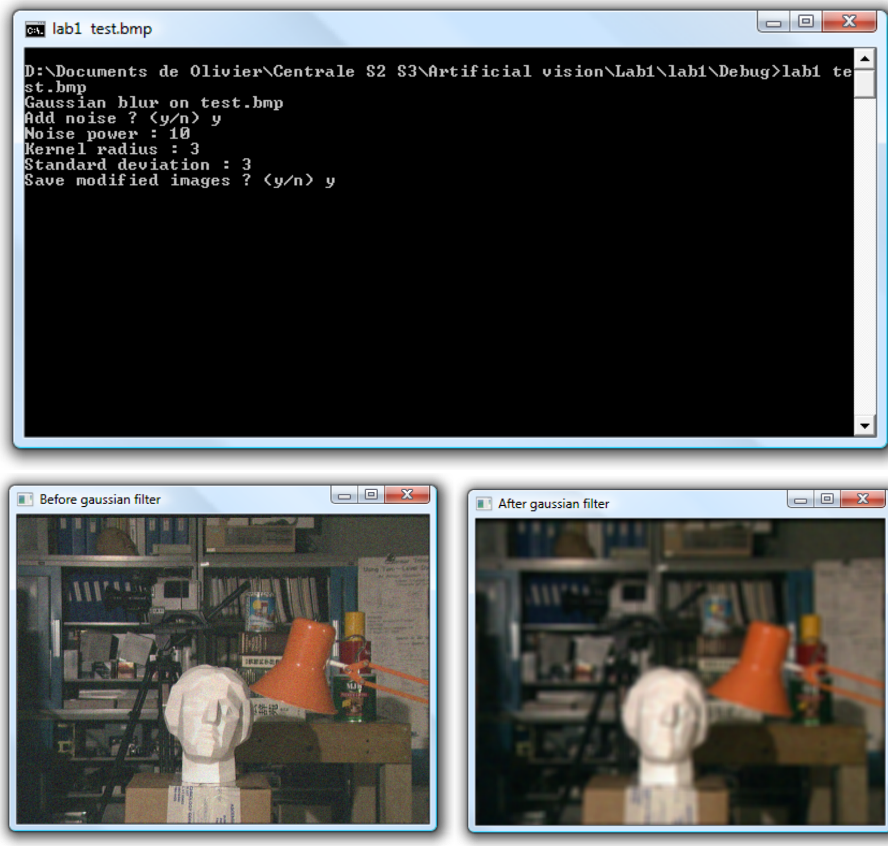
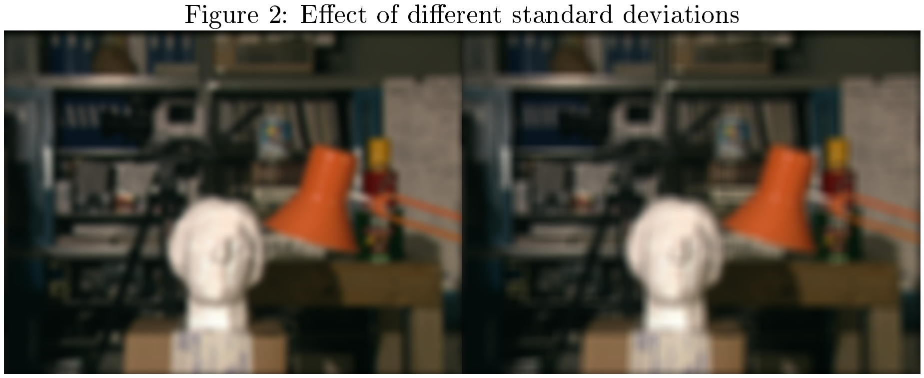


Figure 2: Effect of different standard deviations



The left image as been filtered by a kernel with higher standard deviation than the right one.

Figure 3: Standard deviation higher than radius

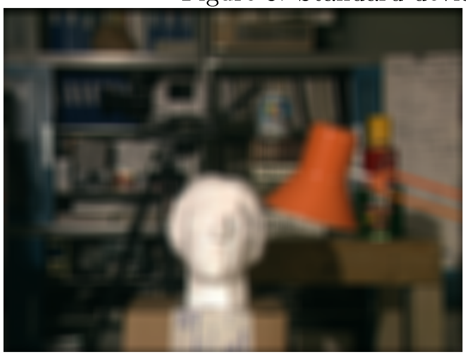


Figure 4: Gaussian denoising

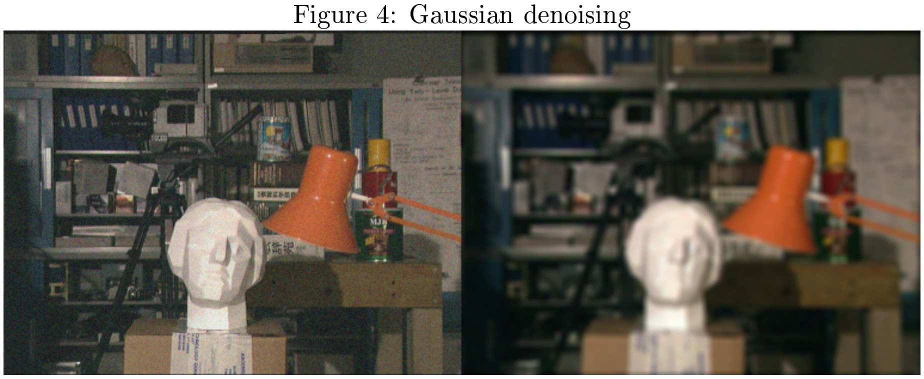


Figure 5: Gaussian effect on very noisy image