영상처리 2차 과제

컴퓨터 공학과 17101244 조서연

첨부 압축파일(20개의 이미지)를 과제에 사용하는 것이 과제의 내용이다. 다수의 노이즈를 가진 이미지를 가지고 원본의 이미지를 추출해내는 방법으로는 ''Arithmetic Operations"이 있다.

$$g(x,y) = f(x,y) + \eta(x,y)$$

» $g(x,y)$: a corrupted image formed by the addition of noise to noiseless image

» $f(x,y)$: noiseless image / $\eta(x,y)$: noise

$$\bar{g}(x,y) = \frac{1}{K}\sum_{i=1}^{K} g_i(x,y)$$

then,

$$E\{\bar{g}(x,y)\} = f(x,y)$$

$$\sigma^2_{\bar{g}(x,y)} = \frac{1}{K}\sigma^2_{\eta(x,y)}$$

» $E\{\bar{g}\}$: the expected value of $\bar{g}$

» $\sigma^2_{\bar{g}(x,y)}$ and $\sigma^2_{\eta(x,y)}$: the variance of $\bar{g}$ and η, respectively

위 식은 노이즈가 첨가된 영상과, 원본 영상, 노이즈간의 관계를 나타낸다. 식의 의미를 살펴보면 이렇다. 노이즈가 낀 영상들의 평균은 원본 영상이라는 것이다. 다만, 이 경우 노이즈가 낀 영상이 1 개, 2 개 있다고 해서 되는 것이 아니고, 많은 노이즈가 낀 영상들이 필요하다. 그렇게 되면, 노이즈끼리의 상충작용이 일어나서, 원본영상에 가까워 진다는 것이다. 그리고 노이즈가 낀 영상들의 평균은 원본영상 이었으므로, 당연히, 노이즈가 낀 영상들은, 원본영상을 중심으로 노이즈만큼 차이가 날테니까, 노이즈가 낀 영상들의 분산은 노이즈들의 분산들과 같다는 것을 알 수 있다. 이렇게, 노이즈가 낀 영상들 수십개를 평균 내서, 노이즈가 없는 원본영상을 찾으려고 하는 과정을 **영상 평균화(image averaging)** 이라 한다. 이번 과제는 20 개의 노이즈가 낀 이미지를 활용하여 원본 이미지를 생성해 내는 과제이다. 이를 영상 평균화(image averaging)을 활용하여 찾은 원본 이미지는 다음 페이지의 오른쪽 그림인 result 와 같다.

그림 1 lenna_gray_00.png

그림 2 lenna_result.png

주어진 20개의 이미지를 영상 평균화 시킨 lenna_result.png와 비교 이미지인 lenna_gray_00.png 이다. 영상 평균화 코드는 아래와 같다.

```
void imageAdd(Mat& image1, Mat& image2, Mat& result)
{
    int numOfLines = image1.rows; // number of lines in the image
    int numOfPixels = image1.cols; // number of pixels per a line
    result.create(image1.rows, image1.cols, image1.type());
    for (int r = 0; r < numOfLines; r++)
    {
        const uchar* data1_in = image1.ptr<uchar>(r);
        const uchar* data2_in = image2.ptr<uchar>(r);
        uchar* data_out = result.ptr<uchar>(r);
        for (int c = 0; c < numOfPixels; c++)
        {
            data_out[c] = saturate_cast<uchar>((data1_in[c] + data2_in[c])/2);
        }
    }
}
```

이미지를 Mat 형식으로 받아 image의 행 수를 rows, 열 수를 cols로 두어 rows*cols(512*512) 형태의 Matrix를 구성하며 각각의 픽셀 위치에 해당하는 BGR 값이 할당되어 있다. 두개의 이미지를 받아 해당 위치의 픽셀의 색의 평균을 data_out, 즉 result에 저장한다.