수치해석

HW03-Adaptive Interpolation

신윤석

2019202053

컴퓨터정보공학부

Introduction

이번 과제는 저번 과제와 마찬가지로 사진의 해상도를 높이는 과제인데, Least Square Approximation 방법을 이용해서 7*7 필터를 만들어서 interpolation을 수행해 해상도를 높인다. 이때, 각 픽셀을 활동성과 방향성 지표로 분류를 해서 필터를 분류에 맞게 만들고 적용해서 노이즈를 줄인다.

Implementation

과제를 수행함에 있어서 수행해야 할 작업을 나열하자면, 픽셀 분류하기 -> 필터 만들기, 픽셀 분류하기 -> 필터 적용하기 순서로 작업을 해야 한다. 결론적으로 필요한 작업은 픽셀 분류하기, 필터 만들기, 필터 적용하기이다.

우선 픽셀을 분류하는 방법은 과제명세서에 나와있는 방법을 따른다. 필터를 통과한 값을 기준으로 픽셀을 나누게 된다.

1	2	2	0	-2	-2	-1	-1	-1	-1	-1	-1	-1	-1	0	3	2	1	1	1	1	-1	-1	-1	-1	-2	-3	0
1	2	3	0	-3	-2	-1	-2	-2	-3	-4	-3	-2	-2	-3	0	4	3	2	1	1	-1	-1	-2	-3	-4	0	3
1	3	4	0	-4	-3	-1	-2	-3	-4	-5	-4	-3	-2	-2	-4	0	5	4	2	1	-1	-2	-4	-5	0	4	2
1	4	5	0	-5	-4	-1	0	0	0	0	0	0	0	-1	-3	-5	0	5	3	1	-1	-3	-5	0	5	3	1
1	3	4	0	-4	-3	-1	2	3	4	5	4	3	2	-1	-2	-4	-5	0	4	2	-2	-4	0	5	4	2	1
1	2	3	0	-3	-2	-1	2	2	3	4	3	2	2	-1	-1	-2	-3	-4	0	3	-3	0	4	3	2	1	1
1	2	2	0	-2	-2	-1	1	1	1	1	1	1	1	-1	-1	-1	-1	-2	-3	0	0	3	2	1	1	1	1
Vertical							Horizontal							Diagonal 135							Diagonal 45						

위 필터를 거친 값 4개 중 Vertical값과 Horizontal값의 유클리드 거리의 제곱을 구해서 각 픽셀의 활동성을 구한다. 이때, 5개의 구간으로 양자화를 해야 하는데 lloyd-max 양자화 방법을 통해 양자화를 하게 된다. 과제에서는 100번 반복해서 구한 구간을 취한다.

위 필터를 거친 값 4개 중 가장 큰 값을 방향성으로 정한다. 그런데, 방향성이 없는 경우가 있을 수 있다. 가장 쉬운 경우는 네 값 모두 작은 경우라서 노이즈로 취급할 수 있는 정도인 경우에 방향성이 없는 것으로 볼 수 있다. 하지만, 그러면 활동성이 큰데 방향성이 없는 것이 설명이 되지 않는다. 결국 두가지를 생각 해 볼 수 있다. 우선 네 값 중 가장 큰 값을 골라야 한다. 통계적인 방법으로는 네 값의 평균과 표준편차를 구해서 p값을 조정해서 고른 표본이 지정된 p값을 넘기는 표본이었으면 방향성이 있는 걸로 인정하고 못 넘기면 방향성이 없는 것으로 생각할 수 있다. 다른 방법은 벡터를 보는 관점에서 생각해 보면 활동성 벡터의 크기보다 단일 방향 벡터의 크기가 더 크다면 해당 방향으로 방향성이 있다고 인정하는 것이다. 하지만 그렇게 되면 Vertical과 Horizontal 방향으로는 방향으로 인정되기 힘들다. 이미 활동성 벡터가 두 방향의 유클리드 거리를 계산한 것이기 때문에 활동성 벡터가 Horizontal방향이나 Vertical 방향으로 완전히 겹쳐 있지 않는 한 Vertical 방향과 Horizontal 방향은 인정되지 않을 것이다. 이런 문제를 해결하기 위해

서 임의의 값을 곱해서 값을 어느정도 조정해서 방향성 여부를 인정하는 방법을 생각 할 수 있다. 실제로 구현하기 위해서는 통계적인 방법에서는 0.5sigma보다 표본의 값이 크면 방향성으로 인정하고, vector구현에서는 방향성 벡터의 크기의 $\sqrt{2}$배를 취한 값과 활동성 백터의 크기 값을 비교했을 때 전자가 크면 방향성으로 인정해줬다. 결과적으로 벡터를 취해서 계산하는 쪽이 0.05~0.1 dB 정도 PSNR이 좋았다.

다음은 필터를 만드는 과정이다. 필터는 선형대수학적 연산을 통해 만들어진다. 우선 해상도를 올리고 싶은 이미지의 일부분을 X라고 하고, 필터를 F, X를 필터에 통과시킨 결과를 Y′이라고 하자. 이때, F는 필터로서 49개의 스칼라를 가진 열벡터이고, X는 필터에 통과시키고 싶은 이미지의 일부분을 각 행으로 쌓은 n*49사이즈의 행렬이다. 그러면 그 결과물인 Y′도 열백터로 나올 것이다.

$$\boldsymbol{Y'} = \boldsymbol{XF}$$

Y′의 목표 값인 ground-truth값을 Y라고 상정하고 이 열벡터는 Y′에 대응하는 목표 값이 담겨있는 벡터라고 생각해보자. 그럼 그 오차 L은 다음과 같을 것이다.

$$L = \|\boldsymbol{Y} - \boldsymbol{Y'}\|^2 = \|\boldsymbol{Y} - \boldsymbol{XF}\|^2$$

L이 최소가 되게 하는 F를 찾고 싶은 것이니 F로 미분해서 0이되는 F를 찾으면 된다. 그 F가 최적화된 필터이다. 다만 선형방정식이므로 기호가 조금 다르다.

$$\nabla_{\mathbf{F}} L = \nabla_{\mathbf{F}} \|\boldsymbol{Y} - \boldsymbol{XF}\|^2 = 2\boldsymbol{X}^T\boldsymbol{XF} - 2\boldsymbol{X}^T\boldsymbol{Y} = \mathbf{0}$$
$$\boldsymbol{X}^T\boldsymbol{XF} = \boldsymbol{X}^T\boldsymbol{Y}$$
$$\therefore \boldsymbol{F} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{Y}$$

구하고자 하는 것이 F였으므로 결국 X와 Y에 대한 연산을 취하면 원하는 필터 F를 구할 수 있다. 이번 과제에서는 역행렬을 구하기 위한 알고리즘으로 가우스-요르단 소거법을 이용하여 구했다. 수치해석상의 오차를 극복하기 위해 pibot을 상대크기가 가장 큰 수를 갖는 열을 선택해서 골랐다. 사실, 정확히는 역행렬을 구한 것이 아니라 F를 구하기 위해서 그냥 행렬로 된 연립방정식인 $\boldsymbol{X}^T\boldsymbol{XF} = \boldsymbol{X}^T\boldsymbol{Y}$를 푼 것이다. 이때, 계산 량을 줄이기 위해 열행렬 Y를 H픽셀, V픽셀, D픽셀 위치에 있는 ground-truth 데이터를 n*3크기의 행렬로 모두 써서 한번에 계산하였다. 이러면 같은 역행렬을 3번 구할 필요 없이 한번 구한 역행렬을 3번 이용해서 각 F를 찾은 것과 동일한 효과가 난다.

필터를 구했으므로, 필터를 구하기 위해 픽셀의 유형을 구분했던 방법 그대로 다시 해상도를 높이고 싶은 이미지의 픽셀을 구분한다. 그리고 필터를 적용해주면 interpolation이 끝나게 된다. 필터를 적용하는 방법에는 크게 알고리즘 없이 그냥 Multiplicate And aCcumulate 연산을 하면 된다. 참고로, 이번 과제에서 padding은 symmetry padding을 하였다. 이미지 끝에서 픽셀 데이터가 없을 때, 해당 위치에서 대칭된 위치의 데이터를 가져왔다.

Result

-Lena

Ground truth

Adaptive Interpolated

PSNR: 11.871655 dB

Bilinear interpolated

PSNR: 11.040975 dB

-Couple

Ground truth

Adaptive Interpolated

PSNR: 7.710498 dB

Bilinear

PSNR: 7.44935 dB

-Barbara

Ground truth

Adaptive Interpolated

PSNR: 8.236774

Bilinear

PSNR: 7.364886

Conclusion

전체적인 결과를 봤을 때, Adaptive interpolation을 적용한 영상이 대조군으로 삽입한 Bilinear보다 PSNR이 근소하게 높았다. 대략 0.26~0.88 dB정도 더 높았다. 그러나 눈으로 보는 차이는 더 크다고 느껴진다. Bilinear같은 경우는 육안으로 볼 때, 결과물이 약간 블러처리가 된 것이 느껴지고, 특히 barbara에서 대각선 무늬에 다른 무늬도 추가된 점이 눈에 띈다. 물론 결과물이 블러처리된 것처럼 보이는 이유는 저해상도로 만들 때, anti-alliancing을 줄이기 위해 LPF를 적용 시킨 이미지라는 것을 감안하면 오히려 블러처리 된 것 같은 이미지가 더 자연스러운 결과라고 생각할 수 있다. 또한, edge에서의 계단 현상이 눈에 띈다. Adaptive Interpolation의 경우에는 high frequency 부분이 어느정도 살아나서 블러처리한 느낌이 좀 덜하거나 거의 없는 수준이었다. 특히나, 계단 현상 없이 부드럽고 원본과 비슷한 느낌이 나는 점이 인상깊었다. 다만, 너무나 특성을 많이 반영한 탓인지 Barbara 사진에서 인물의 머리카락을 보면 2시방향에서 7시 방향으로 머리카락에 인위적인 무늬가 나 있는 부분이 보였다.

결론적으로는 Leaner Square Approximation 기반의 Interpolation이 Bilinear interpolation과 PSNR상으로는 크게 차이가 나지 않지만 사람의 눈으로 볼 때는 크다고 할 수 있을 정도의 차이가 느껴지는 결과물을 얻을 수 있었다.

하지만 아쉬운 점은 이미지를 처리하기 위한 filter를 만드는 과정에서 결국 원본 이미지가 필요했다는 점이다. 머신 러닝처럼 여러 이미지에 대해서 미리 Leaner Square Approximation을 통해 Filter를 구해 놓고 다른 이미지에 필터를 쓴다면 이 과제의 현실적인 응용이 될 것이다.

이상으로 과제를 마치겠습니다.