

※ 각자가 임의로 선택한 영상신호(또는 음성신호)를

- (1) 저주파 통과 필터링을 수행하라.
- (2) 고주파 통과 필터링을 수행하라.

(1) 저주파 통과 필터링

system('convert totoro.jpg totoro.pgm');	%jpg이미지를 pgm이미지로 변환
x=imread('totoro.pgm');	%이미지를 읽음
imshow(x)	%이미지를 출력
[YY, XX] = size(x)	%이미지의 사이즈를 구함
X=fftshift(fft2(x));	%이미지를 fft 시킴
image1 = mat2gray(log(1+ abs(X)));	%mat2gray로 픽셀값에 맞게 정규화 시킴 log0은 에러가 날 수있으므로 1을 더함
figure	%여러 이미지를 출력시키기 위해
imshow(image1)	%이미지를 출력
figure	
mesh(image1)	%이미지를 삼차원 그래프로 출력
R=50;	%필터의 통과 구간을 정함
Yc=round(YY/2);	%이미지의 세로축 가운데를 찾음
Xc=round(XX/2);	%이미지의 가로축 가운데를 찾음
H=zeros(YY, XX);	%만들 필터 값을 모두 0으로 만들
H(Yc-R:Yc+ R, Xc-R:Xc+ R)=1;	%필터 통과 부분을 1로 만들
figure	
mesh(H)	%필터를 삼차원 이미지로 출력
result = X .* H;	%X에 고주파 필터를 통과시킴
result1 = mat2gray(log(1+ abs(result)));	%필터 통과 이후 grayscale이미지로 만들어 result1에 넣음
figure	
imshow(result1)	%grayscale시킨 이미지를 출력
result2=ifft2(fftshift(result));	%필터통과 결과를 ifft시켜 복원
image2=mat2gray(abs(result2));	%정규화 시킨 뒤 image2에 넣음
figure	
imshow(image2)	%이미지를 출력
입력영상	

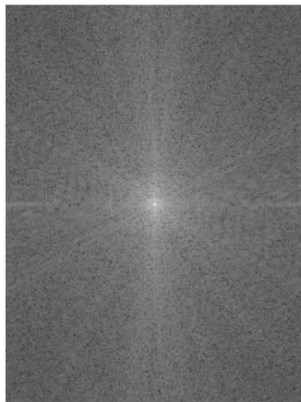
File Edit



A | G | P | R | ? | [84.57, 443.3]

입력 이미지를 fft2 한 결과의 gray scale 이미지

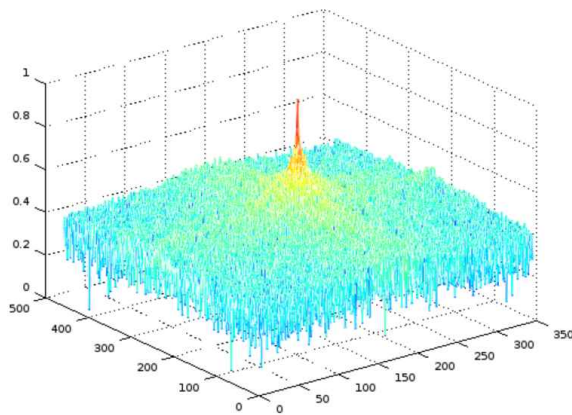
File Edit



A | G | P | R | ? | [69.79, 124.8]

입력 이미지를 fft2 한 결과의 삼차원 그래프

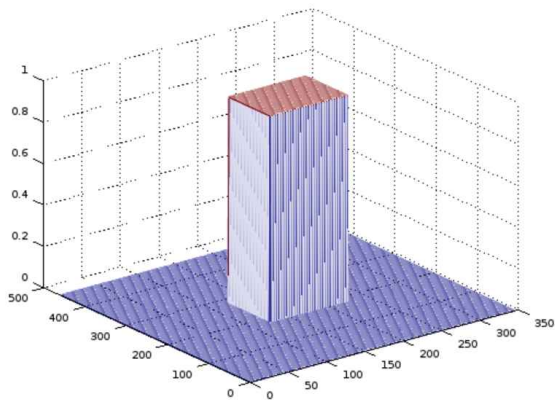
File Edit



A | G | P | R | ? | [290.5, -183]

저주파 필터의 삼차원 그래프

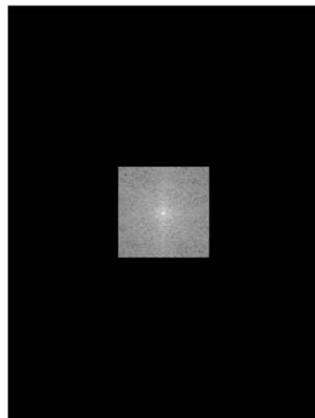
File Edit



A|G|P|R|? [69.25, 55.25]

입력 이미지를 fft 변환시킨 결과에 저주파 필터를 실행한 결과

File Edit



A|G|P|R|? [127.6, -8.235]

저주파 필터링 이미지

File Edit




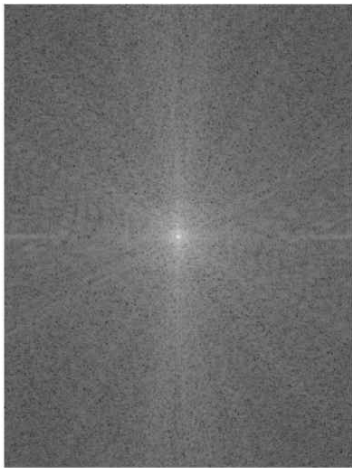
A|G|P|R|? [147.7, 303.5]

설명

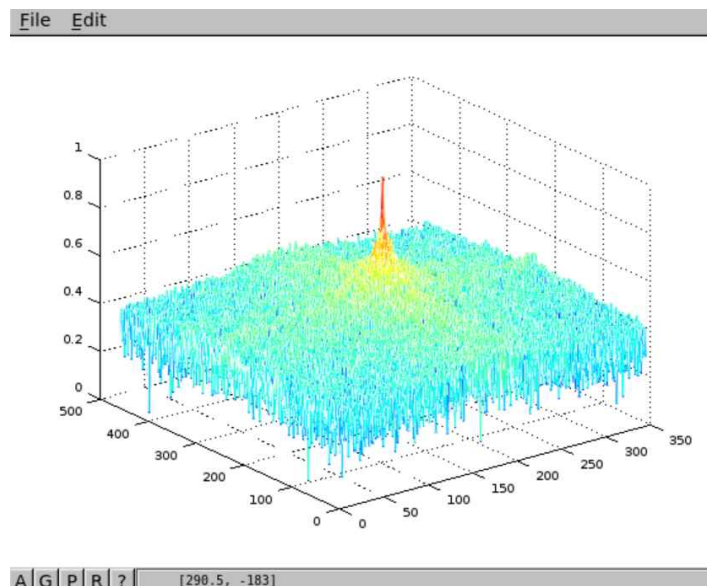
Xftp4를 사용하여 네이버에서 다운받은 토토로 이미지를 서버에 올렸습니다.
system('convert totoro.jpg totoro.pgm')를 사용하여 jpg이미지를 pgm이미지로 변환시켰습니다.
[YY, XX] = size(x)를 사용하여 이미지의 세로와 가로 사이즈를 구했습니다.
fftshift(fft2(x))에서 fftshift 는 0을 가운데에 둬서 좌우 방향 데이터를 분석하기 쉽게 도와주고 fft2(고속푸리에변환)는 2차원 이미지를 빠르게 이산 푸리에 변환 시킵니다.
mat2gray(log(1+abs(X)))에서 mat2gray() 함수는 이미지를 흑백이미지로 만들고 픽셀 값을 (0~255)로 정규화 시킵니다. log를 사용하여 너무 큰 값을 비교적으로 많이 줄여 편차가 작게 만들어줍니다. log(0)이 될 경우 무한대가 되어 에러가 날 수 있으므로 1을 더해 줍니다. R의 값을 설정하므로서 H(Yc-R:Yc+R, Xc-R:Xc+R)=1에서 필터의 크기를 결정할 수 있습니다. R을 크게 할수록 필터의 크기가 커집니다. Yc=round(YY/2), Xc=round(XX/2)를 사용하여 이미지의 중간을 구합니다. H=zeros(YY, XX)를 사용하여 입력 이미지와 같은 크기의 필터 배경을 0으로 만들고 H(Yc-R:Yc+R, Xc-R:Xc+R)=1 사용하여 중간부터 상하좌우 R만큼을 1로 만들어 필터를 만들어 줍니다. result = X .* H에서 입력 이미지를 fft시킨 것에 필터링을 해줍니다. result2=ifft2(fftshift(result))에서는 fft시켰던 결과를 다시 복원시킵니다. image2=mat2gray(abs(result2))에서 복원한 결과를 정규화 시켜 입력영상에 필터링을 적용한 영상을 image2에 넣습니다.

(2) 고주파 통과 필터링

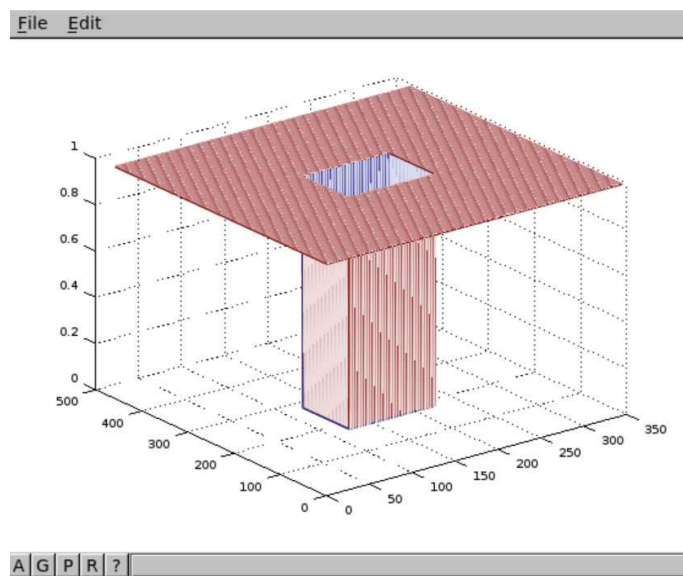
system('convert totoro.jpg totoro.pgm');	%jpg이미지를 pgm이미지로 변환
x=imread('totoro.pgm');	%이미지를 읽음
imshow(x)	%이미지를 출력
[YY, XX] = size(x)	%이미지의 사이즈를 구함
X=fftshift(fft2(x));	%이미지를 fft 시킴
image1 = mat2gray(log(1+abs(X)));	%mat2gray로 픽셀값에 맞게 정규화 시킴 log0은 에러가 날 수있으므로 1을 더함
figure	%여러 이미지를 출력시키기 위해
imshow(image1)	%이미지를 출력
figure	
mesh(image1)	%이미지를 삼차원 그래프로 출력
R=50;	%필터의 통과 구간을 정함
Yc=round(YY/2);	%이미지의 세로축 가운데를 찾음
Xc=round(XX/2);	%이미지의 가로축 가운데를 찾음
H=ones(YY, XX);	%만들 필터 값을 모두 1로 만들
H(Yc-R:Yc+R, Xc-R:Xc+R)=0;	%필터 통과 부분을 0으로 만들
figure	
mesh(H)	%필터를 삼차원 이미지로 출력
result = X .* H;	%X에 고주파 필터를 통과시킴
result1 = mat2gray(log(1+abs(result)));	%필터 통과 이후 grayscale이미지로

<pre>figure imshow(result1) result2=ifft2(fftshift(result)); image2=mat2gray(abs(result2)); figure imshow(1-image2)</pre>	<p>만들어 result1에 넣음</p> <p>%grayscale시킨 이미지를 출력 %필터통과 결과를 ifft시켜 복원 %정규화 시킨 뒤 image2에 넣음</p> <p>%1-image2를 하여 잘 보이게 색을 반전 시켜 이미지 출력</p>
<p>입력영상</p> 	
<p>입력 이미지를 fft2 한 결과의 gray scale 이미지</p> 	

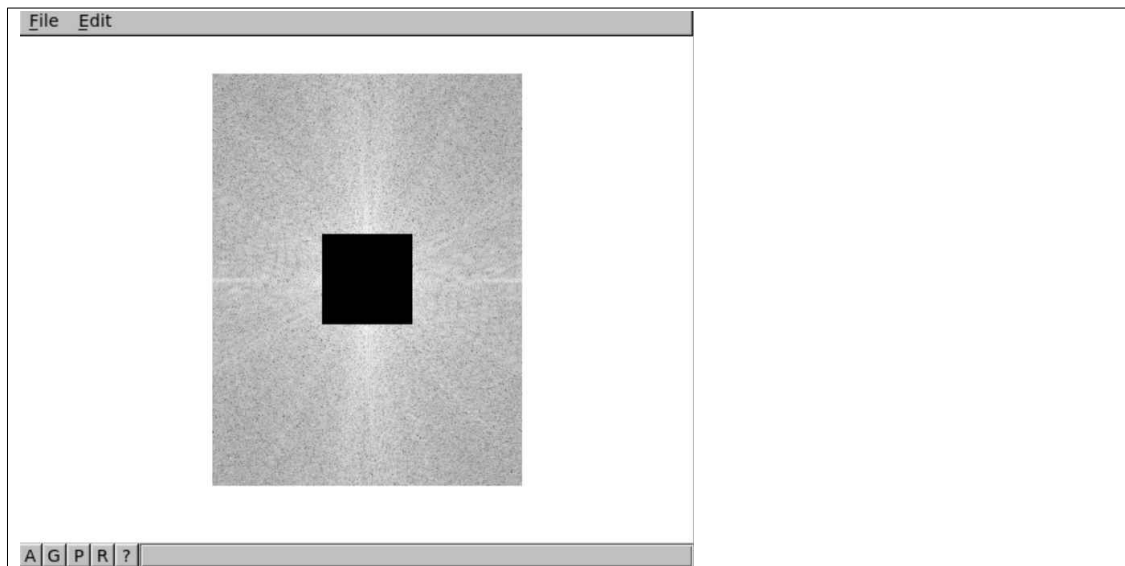
입력 이미지를 fft2 한 결과의 삼차원 그래프



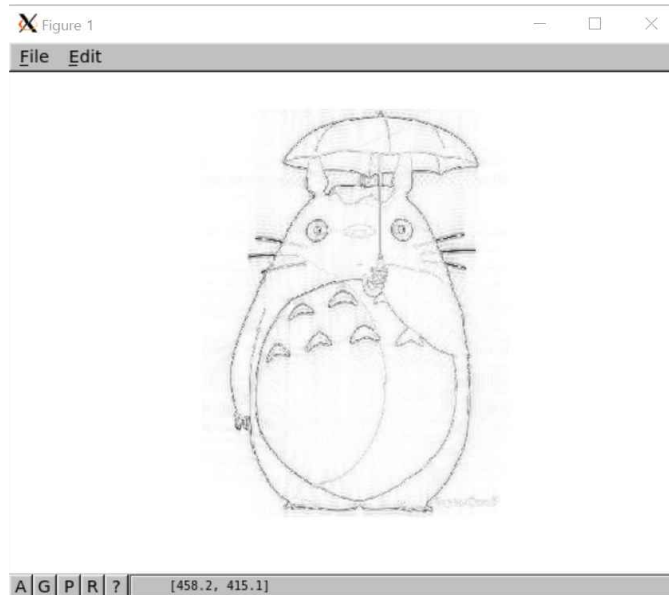
고주파 필터의 삼차원 그래프



입력 이미지를 fft 변환시킨 결과에 저주파 필터를 실행한 결과



고주파 필터링 이미지를 색 반전 시킨 결과



설명

Xftp4를 사용하여 네이버에서 다운받은 토도로 이미지를 서버에 올렸습니다.

system('convert totoro.jpg totoro.pgm')를 사용하여 jpg이미지를 pgm이미지로 변환 시켰습니다.

[YY, XX] = size(x)를 사용하여 이미지의 세로와 가로 사이즈를 구했습니다.

fftshift(fft2(x))에서 fftshift 는 0을 가운데에 둬서 좌우 방향 데이터를 분석하기 쉽게 도와주고 fft2(고속푸리에변환)는 2차원 이미지를 빠르게 이산 푸리에 변환 시킵니다.

mat2gray(log(1+abs(X)))에서 mat2gray() 함수는 이미지를 흑백이미지로 만들고 픽셀 값을 (0~255)로 정규화 시킵니다. log를 사용하여 너무 큰 값을 비교적으로 많이 줄여 편차가 작게 만들어줍니다. log(0)이 될 경우 무한대가 되어 에러가 날 수 있으므로 1을 더해 줍니다. R의 값을 설정하므로써 $H(Y_c-R:Y_c+R, X_c-R:X_c+R)=0$ 에서 필터의 크기를 결정할 수 있습니다. R을 크게 할수록 필터의 크기가 커집니다. $Y_c=\text{round}(YY/2)$,

`Xc=round(XX/2)`를 사용하여 이미지의 중간을 구합니다. `H=ones(YY, XX)`를 사용하여 입력 이미지와 같은 크기의 필터 배경을 1로 만들고 `H(Yc-R:Yc+R, Xc-R:Xc+R)=0`을 사용하여 중간부터 상하좌우 R만큼을 0으로 만들어 필터를 만들어 줍니다. `result = X.* H`에서 입력 이미지를 fft시킨 것에 필터링을 해줍니다. `result2=ifft2(fftshift(result))`에서는 fft시켰던 결과를 다시 복원시킵니다. `image2=mat2gray(abs(result2))`에서 복원한 결과를 정규화 시켜 입력영상에 필터링을 적용한 영상을 `image2`에 넣습니다. `imshow(1-image2)`는 이미지를 잘 보이게 하기 위해서 색을 반전시켰습니다.