

컴퓨터정보통신공학부  
2011253020 이화중

Image Transforming  
- FDCT/IDCT

영상처리 HW #4

# Results

## For the Given 512×512 Image ‘Lena’:

### Perform DCT

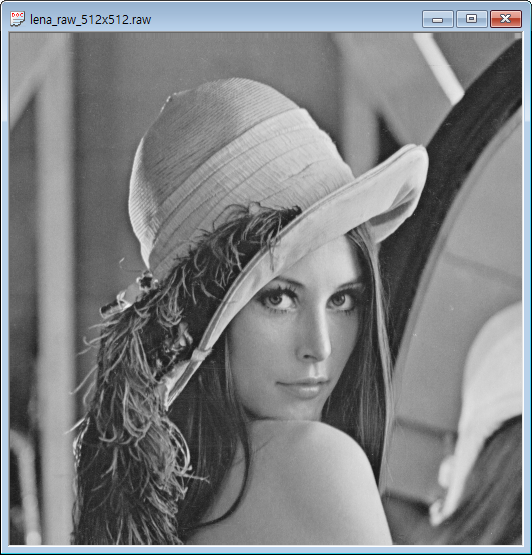


Figure 1 Original Image of ‘Lena’

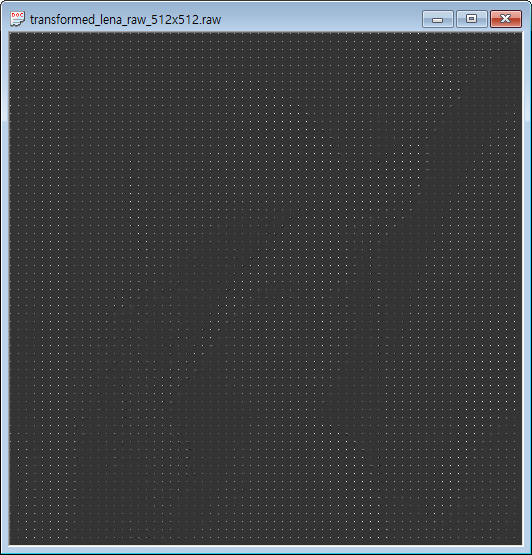


Figure 2 Frequency Spectrum of ‘Lena’ (8×8 Forward DCT)

### Perform IDCT and Compute the MSE



Figure 3 Reconstructed Image of ‘Lena’ (8×8 Inverse DCT)

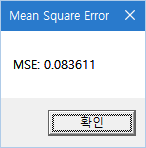


Figure 4 MSE of DCT-Performed ‘Lena’

## For the Given 512×512 Image ‘Boat’:

### Perform DCT

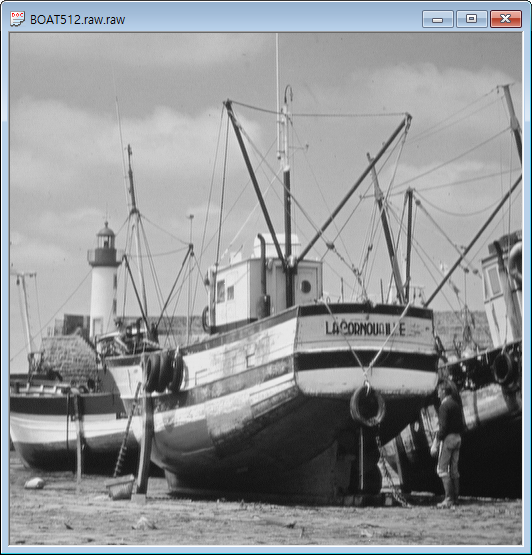


Figure 5 Original Image of ‘Boat’

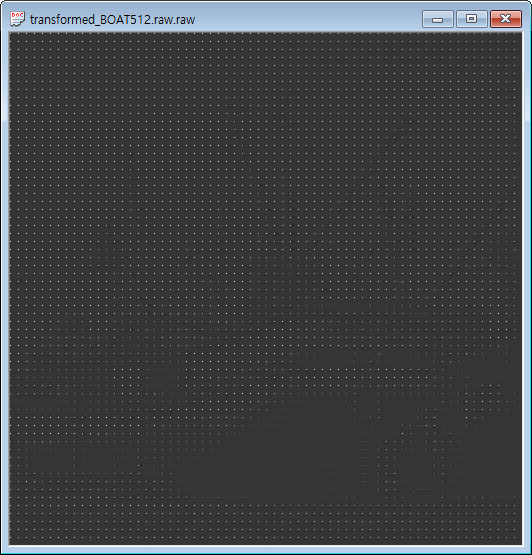


Figure 6 Frequency Spectrum of ‘Boat’ (8×8 Forward DCT)

### Perform IDCT and Compute the MSE

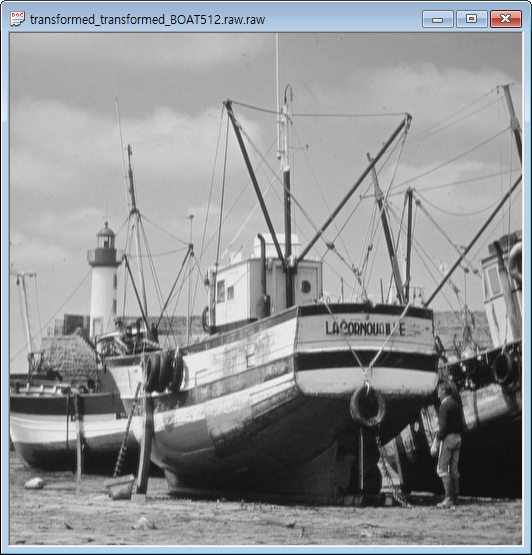


Figure 7 Reconstructed Image of ‘Boat’ (8×8 Inverse DCT)

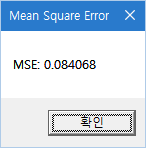


Figure 8 MSE of DCT-Performed ‘Boat’

## Discussion

DCT는 다양한 영상 압축 알고리즘에서 사용된다. 대표적으로 예시로 JPEG를 들 수 있다. DFT와 비교하여, DCT의 장점은 복소수 수준의 계산이 없다는 것이다. 이는 DFT보다 계산 복잡도를 낮추는 동시에, 계산 시간도 단축시킨다.

를 다음과 같다고 할 때,

Forward DCT의 계산식은 다음과 같다:

Inverse DCT의 계산식은 다음과 같다:

DCT의 수행 속도를 개선하기 위한 방법으로는, 행렬 곱과 코사인 함수에 대한 룩업 테이블을 미리 생성하여 두는 방법 등이 있다. 룩업 테이블을 미리 생성하면, 그 계산 시간만큼 단축할 수 있으므로 수행 속도를 향상시킬 수 있다.

‘Lena’ 이미지와 이에 DCT를 적용하고 IDCT로 복원한 이미지의 MSE는 0.083611 (Figure 4)이다. ‘Boat’ 이미지에 대한 MSE는 0.084068 (Figure 8)이다. 이를 통해 ‘Lena’이미지의 픽셀 값이 조금 더 잘 유지되었음을 확인할 수 있다. 각 MSE 값들은 매우 작아, 화질 열화가 적다.

DCT가 무손실 변환 알고리즘임에도 불구하고 MSE 값이 0이 아닌 이유는, 계산상 발생하는 소수점을 제거하기 때문이다. 이 결과로부터, 알고리즘과 달리 실제 계산시에는 데이터의 손실이 발생하였다는 점을 확인할 수 있다. 이같이 손실이 발생함에도 불구하고 DCT 연산은 다양한 영상 처리 분야, 특히 영상압축에 사용된다.

그 이유를 2가지 정도로 압축할 수 있다. 첫 번째 이유는 입력 데이터 성분 사이의 자기 상관성이 높은 경우에 에너지 집중화 성능이 우수하기 때문이다. 다른 변환에 비해 DCT의 성능이 우수하므로 작은 손실은 감수할 수 있다. 이보다도 결정적인 이유는, 인간의 시각적으로 원본 영상과 DCT/IDCT 변환 결과 영상을 비교했을 때 그 차이를 확인하기 어렵다는 점이다. 그 예로, Figure 1과 Figure 3 또는 Figure 6와 Figure 8 을 시각적으로 각각 비교했을 때 그 차이를 거의 느낄 수 없다.

그래도 손실을 없애고 싶다면, FDCT의 결과의 소수점을 제거하지 않고 활용하는 방법을 생각할 수 있다. 하지만, 소수점을 유지하므로 DCT의 결과의 크기가 더 커질 수 밖에 없어, 이를 활용하여 압축했을 때 그 효과가 반감될 수 있다. 게다가, 컴퓨터가 계산할 수 있는 소수점 자릿수에 한계가 있기 때문에, MSE 자체를 0으로 유지하기는 매우 어렵다. 손실을 완전하게 없애는 것은 거의 불가능하다는 것이다.

이러한 DCT는 영상 압축에 활용할 수 있다. DCT의 결과로 주파수 도메인으로 변경되며, 에너지가 큰 저주파 성분이 좌측 상단으로 몰리기 때문이다. 인간의 눈은 고주파에 둔감하므로, 고주파가 손실되더라도 잘 인지하지 못한다. 따라서 중요한 정보인 저주파만 남기고 상대적으로 불필요한 고주파를 제거하면, 압축 효과를 얻을 수 있다.

두 이미지의 에너지 집중화 정도를 비교하면, ‘Boat’ 이미지가 ‘Lena’ 이미지보다 더 좋다. 에너지 집중화 정도는 DCT 결과에 대한 히스토그램을 그려서 시각적으로 확인할 수 있다. 각각의 히스토그램은 다음의 Figure 9와 Figure 10과 같다. 각 히스토그램에서 우측 부분이 각 영상의 저주파에 해당한다. 따라서 이 부분을 비교하는 것으로 에너지 집중화 정도의 비교를 대신할 수 있다. 그 결과, Figure 10의 히스토그램이 Figure 9의 히스토그램보다 우측이 더 몰려 있다. 몰려 있다는 것은 여기에 해당하는 값이 더 많다는 것으로, 집중화가 더 잘 되었다는 것을 의미한다. 따라서 ‘Boat’ 이미지가 ‘Lena’ 이미지보다 에너지 집중화가 잘 되었다.

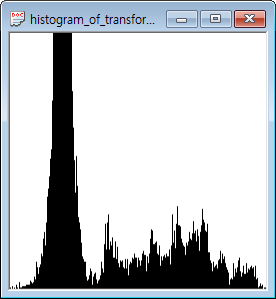


Figure 9 Histogram of DCT-Performed ‘Lena’ (From Figure 2)

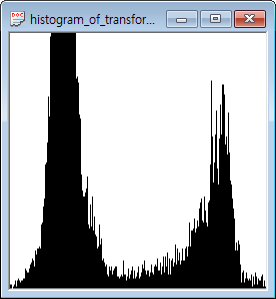


Figure 10 Histogram of DCT-Performed ‘Boat’ (From Figure 6)

원본 영상을 시각적으로 비교해서도 에너지 집중화 정도를 짐작하여 비교할 수 있다. Figure 5는 Figure 1보다 평탄한 부분이 많다. 평탄한 부분은 곧 저주파를 의미한다. 따라서 Figure 5가 Figure 1보다 저주파 성분이 많을 것이라는 것을 짐작할 수 있으며, 이를 통해 에너지 집중화도 잘 될 것이라는 것을 짐작할 수 있다.

# Source Code

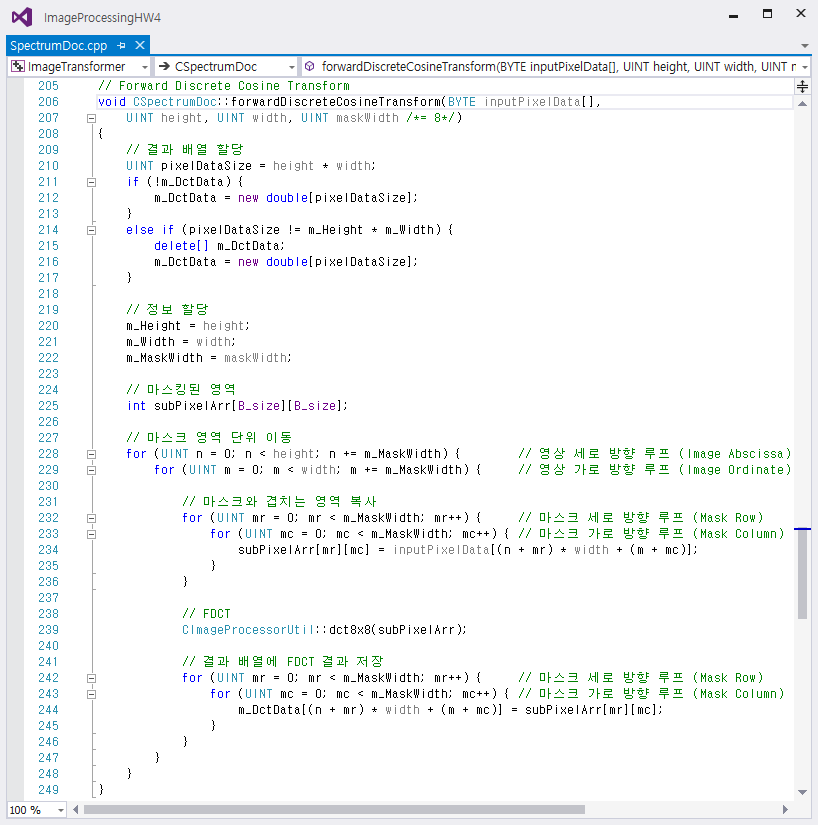


Figure 11 Source Code of FDCT

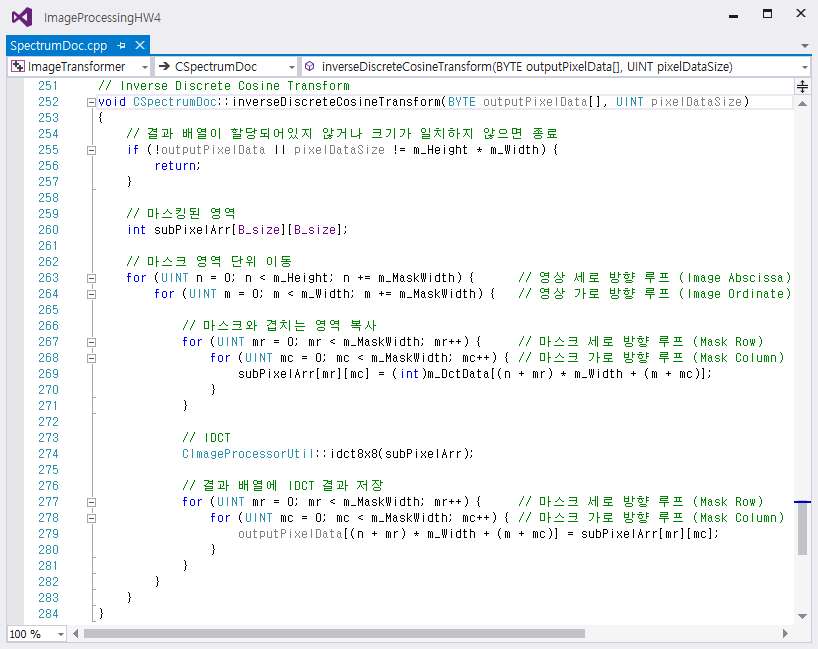


Figure 12 Source Code of IDCT

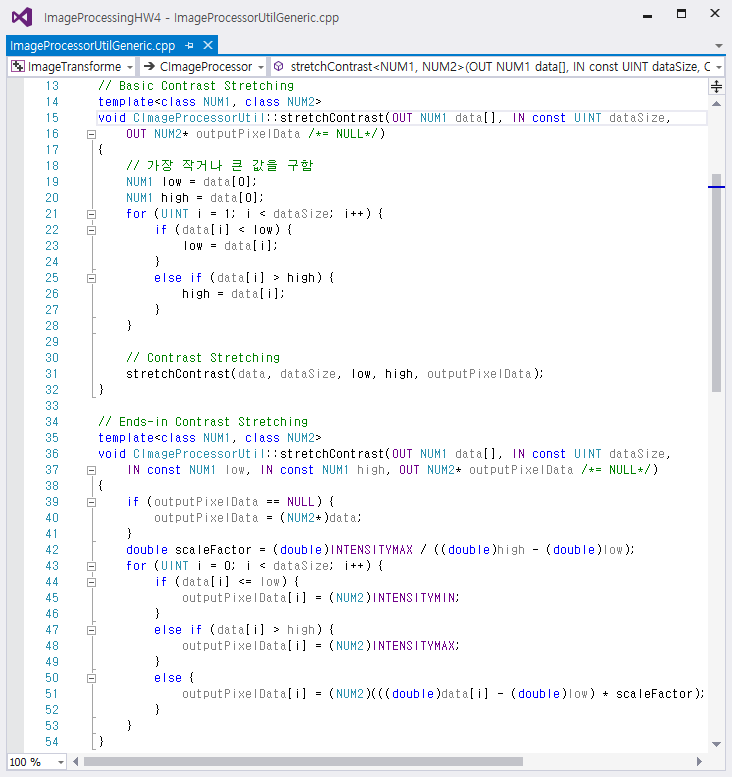


Figure 13 Source Code of Scaling (for DCT Data)

DCT를 수행한 데이터는 [-2048, 2048] 범위에 존재한다. 따라서 이를 시각적으로 화면에서 확인하기 위해서는 [0, 255] 구간으로 스케일링하여야 한다. 원본 데이터는 유지한채 화면 출력시에만 Figure 13의 함수를 호출하여 스케일링하였다. 이 함수는 Template으로 작성하여 범용성을 높였다.

# User Guide

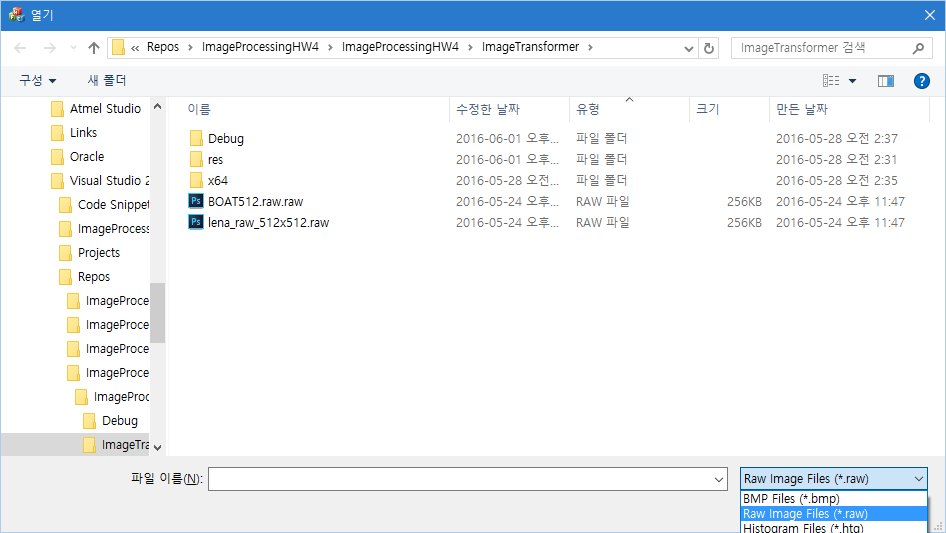


Figure 14 File Open

Raw Image를 열기 위해서는 열기 타입을 “Raw Image Files (\*raw)”로 설정하여야 한다.

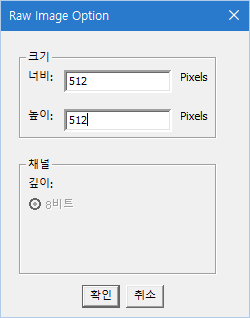


Figure 15 Open Option of Raw Image

Raw 이미지에는 헤더가 없어, 너비와 높이 등에 대한 정보가 없으므로, 사용자가 인위로 지정하여야 한다.

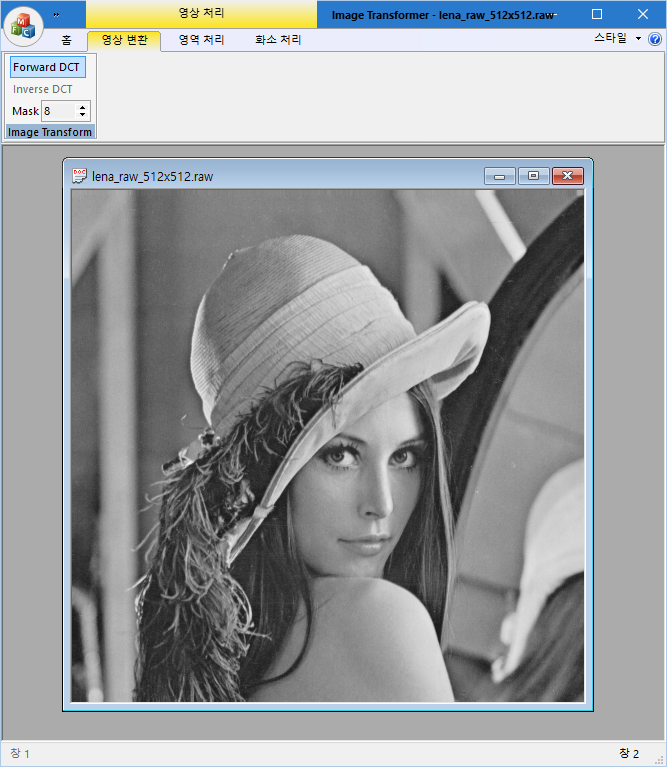


Figure 16 File Opened

이미지를 열면 영상처리 메뉴가 활성화된다. DCT를 수행하기 위해 “Forward DCT”를 클릭한다. 현재 Mask 크기는 지원하지 않는다.

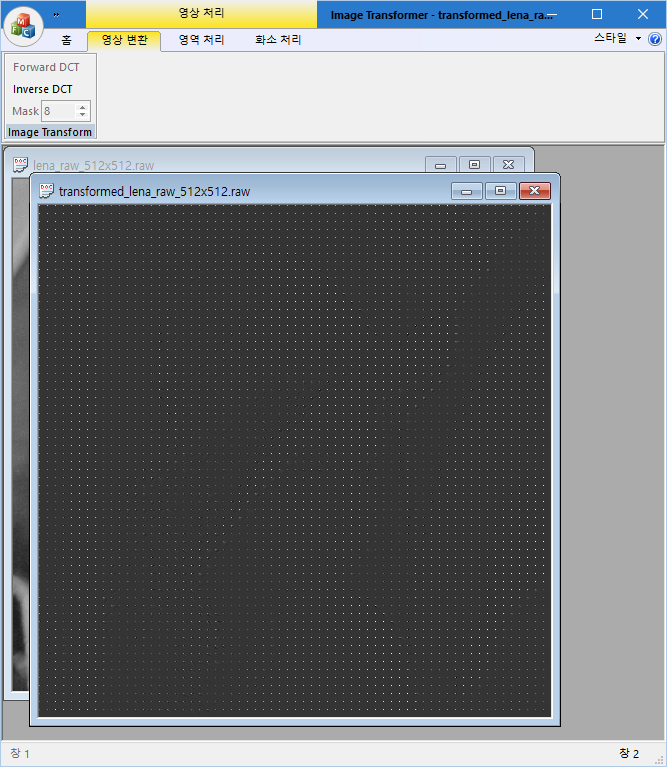


Figure 17 Forward DCT

DCT를 수행하기 위해 “Forward DCT”를 클릭하면 그 결과 스팩트럼이 출력된다. 이 스팩트럼은 시각적으로 확인하기 위하여, 원본 데이터에 모니터 출력에 적합하게 스케일링되어있다.

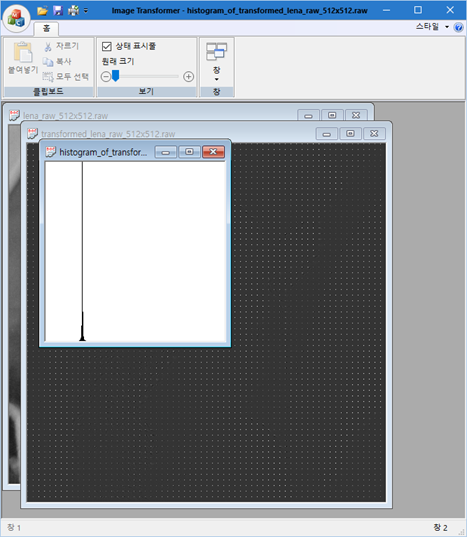


Figure 18 Histogram of FDCT

FDCT의 결과를 활성화 시킨 상태에서 “화소 처리”-“히스토그램 생성”을 클릭하면 히스토그램을 생성한다.

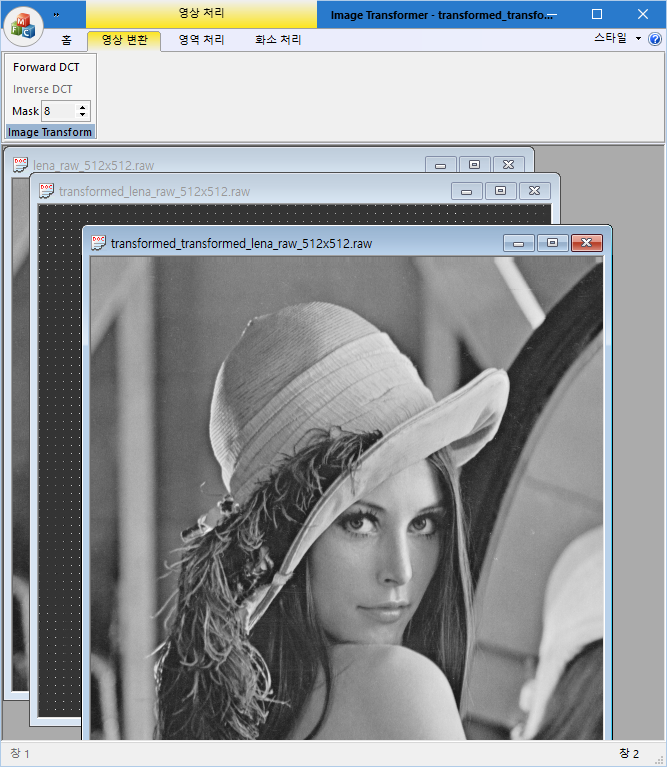


Figure 19 Inverse DCT

FDCT의 결과를 활성화 시킨 상태에서 “Inverse DCT”를 클릭하면 원본 영상과의 MSE를 출력하고 그 결과 영상을 출력한다.

# Environment

Target: Win32 MFC Application

Language: C++

IDE: Microsoft Visual Studio 2015 (v140)

MFC, GDI+