**영상처리 실습 #5**

16010980 이우석

**< 실습 A >**

* Lena



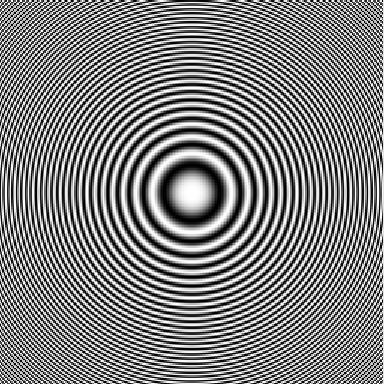
* Prewitt



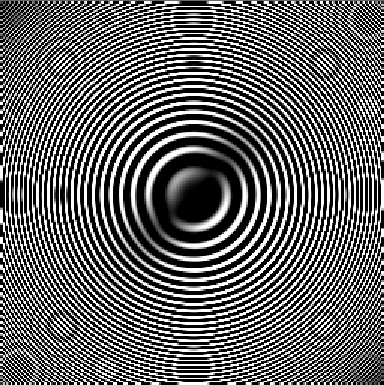
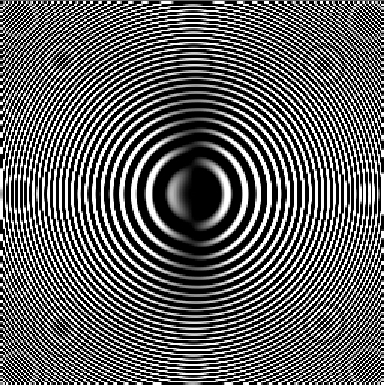
* Sobel

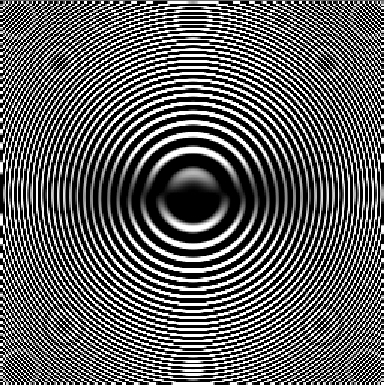


* Zoneplate

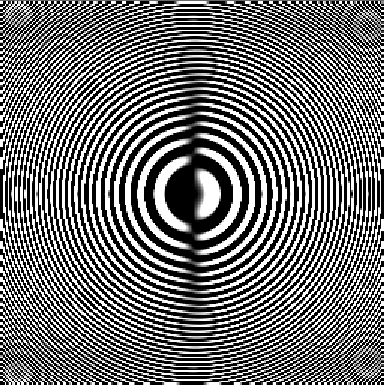


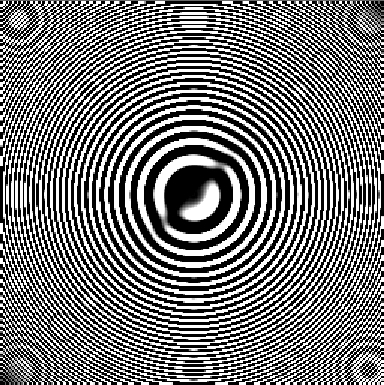
* Prewitt (W, NW, N)



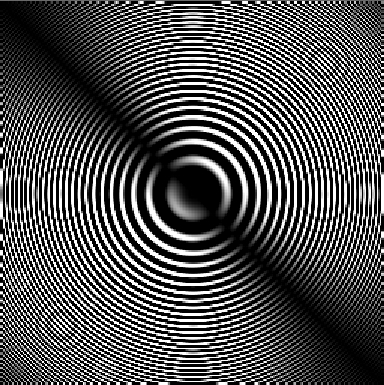
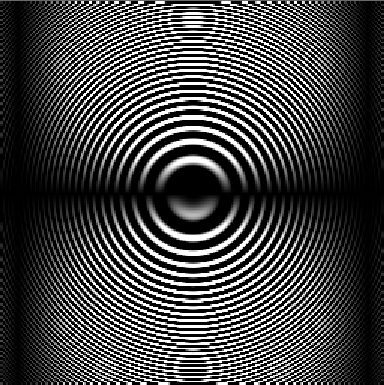


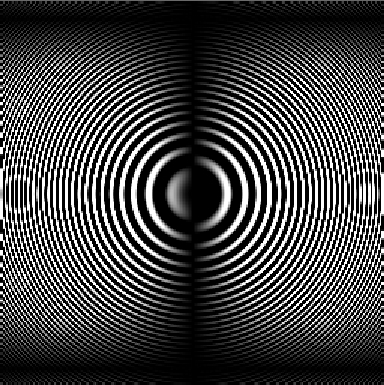
* Kirsh (NE, E, SE)



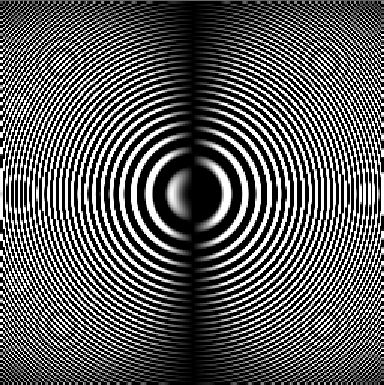
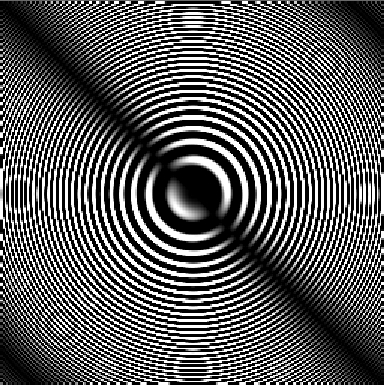
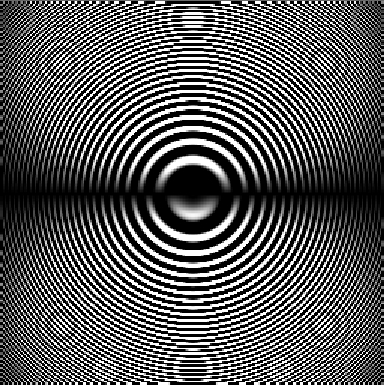


* Robinson 3 (S, SW, W)





* Robinson 5 (S, SW, W)

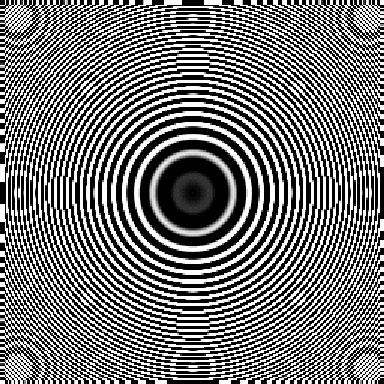
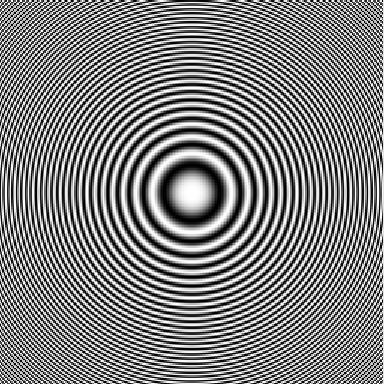


**< 실습 B >**

* Lena



* Zoneplate



**< 실습 C >**

* Lena\_g1, Lena\_g2



* Mean



* Median



* Lena\_s1, Lena\_s2



* Mean



* Median



**< 코드 내용 >**

* LIST3\_1.c

/\* Prewitt \*/

float maskGx\_Prewitt[3][3] = { {-1, -1, -1}, {0, 0, 0}, {1, 1, 1} };

float maskGy\_Prewitt[3][3] = { {-1, 0, 1}, {-1, 0, 1}, {-1, 0, 1} };

/\* Sobel \*/

float maskGx\_Sobel[3][3] = { {-1, -2, -1}, {0, 0, 0}, {1, 2, 1} };

float maskGy\_Sobel[3][3] = { {-1, 0, 1}, {-2, 0, 2}, {-1, 0, 1} };

/\* Compass Prewitt \*/

float mask\_compassPrewitt\_W[3][3] = { {-1, 1, 1}, {-1, -2, 1}, {-1, 1, 1} };

float mask\_compassPrewitt\_NW[3][3] = { {-1, -1, 1}, {-1, -2, 1}, {1, 1, 1} };

float mask\_compassPrewitt\_N[3][3] = { {-1, -1, -1}, {1, -2, 1}, {1, 1, 1} };

/\* Compass Kirsh \*/

float mask\_compassKirsh\_NE[3][3] = { {-3, -3, -3}, {5, 0, -3}, {5, 5, -3} };

float mask\_compassKirsh\_E[3][3] = { {5, -3, -3}, {5, 0, -3}, {5, -3, -3} };

float mask\_compassKirsh\_SE[3][3] = { {5, 5, -3}, {5, 0, -3}, {-3, -3, -3} };

/\* Compass Robinson 3 \*/

float mask\_compassRobinson3\_S[3][3] = { {1, 1, 1}, {0, 0, 0}, {-1, -1, -1} };

float mask\_compassRobinson3\_SW[3][3] = { {0, 1, 1}, {-1, 0, 1}, {-1, -1, 0} };

float mask\_compassRobinson3\_W[3][3] = { {-1, 0, 1}, {-1, 0, 1}, {-1, 0, 1} };

/\* Compass Robinson 5 \*/

float mask\_compassRobinson5\_S[3][3] = { {1, 2, 1}, {0, 0, 0}, {-1, -2, -1} };

float mask\_compassRobinson5\_SW[3][3] = { {0, 1, 2}, {-1, 0, 1}, {-2, -1, 0} };

float mask\_compassRobinson5\_W[3][3] = { {-1, 0, 1}, {-2, 0, 2}, {-1, 0, 1} };

/\* LoG \*/

float mask\_LoG[5][5] = { {0, 0, -1, 0, 0}, {0, -1, -2, -1, 0}, {-1, -2, 16, -2, -1}, {0, -1, -2, -1, 0}, {0, 0, -1, 0, 0} };

/\* detect edge Gx using Prewitt \*/

// convolve(buffer, cols, rows, 3, 3, maskGx\_Prewitt, 0, fileout);

/\* detect edge Gy using Prewitt \*/

// convolve(buffer, cols, rows, 3, 3, maskGy\_Prewitt, 0, fileout);

/\* detect edge Gx using Sobel \*/

// convolve(buffer, cols, rows, 3, 3, maskGx\_Sobel, 0, fileout);

/\* detect edge Gy using Sobel \*/

// convolve(buffer, cols, rows, 3, 3, maskGy\_Sobel, 0, fileout);

/\* detect edge West using Prewitt \*/

// convolve(buffer, cols, rows, 3, 3, mask\_compassPrewitt\_W, 0, fileout);

/\* detect edge NorthWest using Prewitt \*/

// convolve(buffer, cols, rows, 3, 3, mask\_compassPrewitt\_NW, 0, fileout);

/\* detect edge North using Prewitt \*/

// convolve(buffer, cols, rows, 3, 3, mask\_compassPrewitt\_N, 0, fileout);

/\* detect edge NorthEast using Kirsh \*/

// convolve(buffer, cols, rows, 3, 3, mask\_compassKirsh\_NE, 0, fileout);

/\* detect edge East using Kirsh \*/

// convolve(buffer, cols, rows, 3, 3, mask\_compassKirsh\_E, 0, fileout);

/\* detect edge SouthEast using Kirsh \*/

// convolve(buffer, cols, rows, 3, 3, mask\_compassKirsh\_SE, 0, fileout);

/\* detect edge South using Robinson 3 \*/

// convolve(buffer, cols, rows, 3, 3, mask\_compassRobinson3\_S, 0, fileout);

/\* detect edge SouthWest using Robinson 3 \*/

// convolve(buffer, cols, rows, 3, 3, mask\_compassRobinson3\_SW, 0, fileout);

/\* detect edge West using Robinson 3 \*/

// convolve(buffer, cols, rows, 3, 3, mask\_compassRobinson3\_W, 0, fileout);

/\* detect edge South using Robinson 5 \*/

// convolve(buffer, cols, rows, 3, 3, mask\_compassRobinson5\_S, 0, fileout);

/\* detect edge SouthWest using Robinson 5 \*/

// convolve(buffer, cols, rows, 3, 3, mask\_compassRobinson5\_SW, 0, fileout);

/\* detect edge West using Robinson 5 \*/

// convolve(buffer, cols, rows, 3, 3, mask\_compassRobinson5\_W, 0, fileout);

/\* detect edge using LoG \*/

// convolve(buffer, cols, rows, 5, 5, mask\_LoG, 0, fileout);

/\* median and mean filter \*/

// median\_filt(buffer, cols, rows, fileout, 3);

* median\_filt.c

float mean(int\* window, int wsize)

{

int i, sum = 0;

float mean;

quicksort(window, 0, wsize);

for (i = 0; i < wsize; i++)

sum += window[i];

mean = (float)sum / wsize;

return mean;

}

// new\_pixel = mean(window, wsize);