**hist\_func.h**

**1. Code Explanation**

// generate PDF for color image -> R, G, B에 대해 각각 histogram 1개씩

float\*\* cal\_PDF\_RGB(Mat& input) {

int count[L][3] = { 0 };// 0:R, 1:G, 2:B

float\*\* PDF = (float\*\*)malloc(sizeof(float\*) \* L);

for (int i = 0; i < L; i++)

PDF[i] = (float\*)calloc(3, sizeof(float));

// Count

for (int i = 0; i < input.rows; i++) {

for (int j = 0; j < input.cols; j++) {

for (int k = 0; k < 3;k++) {

count[input.at<C>(i, j)[k]][k]++;

}

}

}

// Compute PDF

for (int i = 0; i < L; i++) {

for (int j = 0; j < 3;j++) {

PDF[i][j] = (float)count[i][j] / (float)(input.rows \* input.cols);

}

}

return PDF;

}

// generate CDF for color image

float\*\* cal\_CDF\_RGB(Mat& input) {

int count[L][3] = { 0 };

float\*\* CDF = (float\*\*)malloc(sizeof(float\*) \* L);

for (int i = 0; i < L; i++)

CDF[i] = (float\*)calloc(3, sizeof(float));

////////////////////////////////////////////////

// //

// How to access multi channel matrix element //

// //

// if matrix A is CV\_8UC3 type, //

// A(i, j, k) -> A.at<Vec3b>(i, j)[k] //

// //

////////////////////////////////////////////////

// Count

for (int i = 0; i < input.rows; i++) {

for (int j = 0; j < input.cols; j++) {

for (int k = 0; k < 3;k++) {

count[input.at<C>(i, j)[k]][k]++;

}

}

}

// Compute CDF

for (int i = 0; i < L; i++) {

for (int j = 0; j < 3;j++) {

CDF[i][j] = (float)count[i][j] / (float)(input.rows \* input.cols);

if (i != 0)

CDF[i][j] += CDF[i - 1][j];

}

}

return CDF;

}

**PDF\_CDF.cpp**

**1. Code Explanation**

//plot histogram(PDF)

int hist\_w = 512; int hist\_h = 400;

int bin\_w = cvRound((double)hist\_w / L);

Mat histImage\_PDF(hist\_h, hist\_w, CV\_8UC1, Scalar(0, 0, 0)); //PDF 나타낼 image 초기화

for (int i = 1;i < L;i++) { //line 함수를 이용해 이웃한 두 점을 연결하여 histogram을 그린다

line(histImage\_PDF, Point(bin\_w \* (i - 1), hist\_h - cvRound(PDF[i - 1] \* 16000)),

Point(bin\_w \* i, hist\_h - cvRound(PDF[i] \* 16000)), Scalar(255, 0, 0), 2, 8, 0);

} //그래프의 세로축을 0.025로 맞추기 위해 PDF에 400\*40을 곱한다

//plot histogram(CDF)

Mat histImage\_CDF(hist\_h, hist\_w, CV\_8UC1, Scalar(0, 0, 0)); //CDF 나타낼 image 초기화

for (int i = 1;i < L;i++) {

line(histImage\_CDF, Point(bin\_w \* (i - 1), hist\_h - cvRound(CDF[i - 1] \* 400)),Point(bin\_w \* i, hist\_h - cvRound(CDF[i] \* 400)), Scalar(255, 0, 0), 2, 8, 0);

} //그래프의 세로축을 1로 맞추기 위해 CDF에 400을 곱한다

**2. Results**

**포유류, 여우, 고양이, 보는이(가) 표시된 사진

자동 생성된 설명**

**텍스트이(가) 표시된 사진

자동 생성된 설명**

**hist\_stretching.cpp**

**1. Code Explanation**

// histogram stretching (x1 ~ x2 -> y1 ~ y2)

linear\_stretching(input\_gray, stretched, trans\_func\_stretch, 50, 110, 10, 110);

float\* stretched\_PDF = cal\_PDF(stretched); // stretched PDF

//plot histogram(PDF)

int hist\_w = 512; int hist\_h = 400;

int bin\_w = cvRound((double)hist\_w / L);

Mat histImage\_PDF(hist\_h, hist\_w, CV\_8UC1, Scalar(0, 0, 0)); //PDF 나타낼 image 초기화

for (int i = 1;i < L;i++) { //line 함수를 이용해 이웃한 두 점을 연결하여 histogram을 그린다

line(histImage\_PDF, Point(bin\_w \* (i - 1), hist\_h - cvRound(PDF[i - 1] \*16000)),Point(bin\_w \* i, hist\_h - cvRound(PDF[i] \* 16000)), Scalar(255, 0, 0), 2, 8, 0);

} //그래프의 세로축을 0.025로 맞추기 위해 PDF에 400\*40을 곱한다

//plot histogram(stretched\_PDF)

Mat histImage\_stretched\_PDF(hist\_h, hist\_w, CV\_8UC1, Scalar(0, 0, 0)); //CDF 나타낼 image 초기화

for (int i = 1;i < L;i++) {

line(histImage\_stretched\_PDF, Point(bin\_w \* (i - 1), hist\_h - cvRound(stretched\_PDF[i - 1] \* 16000)),Point(bin\_w \* i, hist\_h - cvRound(stretched\_PDF[i] \* 16000)), Scalar(255, 0, 0), 2, 8, 0);

} //그래프의 세로축을 0.025로 맞추기 위해 PDF에 400\*40을 곱한다

//show histogram

namedWindow("PDF", WINDOW\_AUTOSIZE);

imshow("PDF", histImage\_PDF);

namedWindow("PDF(Stretching)", WINDOW\_AUTOSIZE);

imshow("PDF(Stretching)", histImage\_stretched\_PDF);

void linear\_stretching(Mat& input, Mat& stretched, G\* trans\_func, G x1, G x2, G y1, G y2) {

float constant = (y2 - y1) / (float)(x2 - x1);

// compute transfer function

for (int i = 0; i < L; i++) {

if (i >= 0 && i <= x1)

trans\_func[i] = (G)(y1 / x1 \* i);

else if (i > x1 && i <= x2)

trans\_func[i] = (G)(constant \* (i - x1) + y1);

else

trans\_func[i] = (G)((L - 1 - x2) / (L - 1 - y2) \* (i - x2) + y2);

}

// perform the transfer function

for (int i = 0; i < input.rows; i++)

for (int j = 0; j < input.cols; j++)

stretched.at<G>(i, j) = trans\_func[input.at<G>(i, j)];

}

**2. Results**

**포유류, 여우, 고양이, 보는이(가) 표시된 사진

자동 생성된 설명텍스트, 음악이(가) 표시된 사진

자동 생성된 설명**

**hist\_eq.cpp**

**1. Code Explanation**

hist\_eq(input\_gray, equalized, trans\_func\_eq, CDF); // histogram equalization on grayscale image

float\* equalized\_PDF\_gray = cal\_PDF(equalized); // equalized PDF (grayscale)

//plot histogram(PDF)

int hist\_w = 512; int hist\_h = 400;

int bin\_w = cvRound((double)hist\_w / L);

Mat histImage\_PDF(hist\_h, hist\_w, CV\_8UC1, Scalar(0, 0, 0)); //PDF 나타낼 image 초기화

for (int i = 1;i < L;i++) { //line 함수를 이용해 이웃한 두 점을 연결하여 histogram을 그린다

line(histImage\_PDF, Point(bin\_w \* (i - 1), hist\_h - cvRound(PDF[i - 1] \* 16000)),

Point(bin\_w \* i, hist\_h - cvRound(PDF[i] \* 16000)), Scalar(255, 0, 0), 2, 8, 0);

} //그래프의 세로축을 0.025로 맞추기 위해 PDF에 400\*40을 곱한다

//plot histogram(stretched\_PDF)

Mat histImage\_equalized\_PDF(hist\_h, hist\_w, CV\_8UC1, Scalar(0, 0, 0)); //CDF 나타낼 image 초기화

for (int i = 1;i < L;i++) {

line(histImage\_equalized\_PDF, Point(bin\_w \* (i - 1), hist\_h -cvRound(equalized\_PDF\_gray[i - 1] \* 16000)),Point(bin\_w \* i, hist\_h - cvRound(equalized\_PDF\_gray[i] \* 16000)), Scalar(255, 0, 0), 2, 8, 0);

} //그래프의 세로축을 0.025로 맞추기 위해 PDF에 400\*40을 곱한다

**2. Results**

**고양이, 포유류, 여우, 보는이(가) 표시된 사진

자동 생성된 설명음악이(가) 표시된 사진

자동 생성된 설명**

**hist\_eq\_RGB.cpp**

**1. Code Explanation**

void hist\_eq\_Color(Mat& input, Mat& equalized, G(\*trans\_func)[3], float\*\* CDF) {

// compute transfer function=

//0:B, 1:G, 2:R

for (int i = 0; i < L; i++) {

for (int j = 0;j < 3;j++) {

trans\_func[i][j] = (G)((L - 1) \* CDF[i][j]);

}

}

// perform the transfer function

for (int i = 0; i < input.rows; i++) {

for (int j = 0; j < input.cols; j++) {

for (int k = 0;k < 3;k++) {

equalized.at<C>(i, j)[k] = trans\_func[input.at<C>(i, j)[k]][k];

}

}

}

}

// equalized PDF (RGB)

float\*\* equalized\_PDF\_RGB = cal\_PDF\_RGB(equalized\_RGB);

for (int i = 0; i < 3; i++) {

for (int j = 0;j < L;j++) {

// write PDF

fprintf(f\_PDF\_RGB, "%d\t%f\n", j, PDF\_RGB[j][i]);

fprintf(f\_equalized\_PDF\_RGB, "%d\t%f\n", j, equalized\_PDF\_RGB[j][i]);

// write transfer functions

fprintf(f\_trans\_func\_eq\_RGB, "%d\t%d\n", j, trans\_func\_eq\_RGB[j][i]);

}

}

//plot histogram(PDF)

Mat histImage\_PDF\_B(hist\_h, hist\_w, CV\_8UC1, Scalar(0, 0, 0)); //Blue PDF histogram

Mat histImage\_PDF\_G(hist\_h, hist\_w, CV\_8UC1, Scalar(0, 0, 0)); //Green PDF histogram

Mat histImage\_PDF\_R(hist\_h, hist\_w, CV\_8UC1, Scalar(0, 0, 0)); //Red PDF histogram

//plot histogram(equalized\_PDF\_RGB)

Mat histImage\_equalized\_PDF\_RGB\_B(hist\_h, hist\_w, CV\_8UC1, Scalar(0, 0, 0)); //Blue PDF histogram

Mat histImage\_equalized\_PDF\_RGB\_G(hist\_h, hist\_w, CV\_8UC1, Scalar(0, 0, 0));//Green PDF histogram

Mat histImage\_equalized\_PDF\_RGB\_R(hist\_h, hist\_w, CV\_8UC1, Scalar(0, 0, 0));//Red PDF histogram

**2. Results**

**포유류, 고양이, 실내, 여우이(가) 표시된 사진

자동 생성된 설명**

**텍스트이(가) 표시된 사진

자동 생성된 설명**

**음악, 오르간이(가) 표시된 사진

자동 생성된 설명**

**hist\_eq\_YUV.cpp**

**1. Code Explanation**

// RGB -> YUV

cvtColor(input, equalized\_YUV, CV\_RGB2YUV);

// split each channel(Y, U, V)

Mat channels[3];

split(equalized\_YUV, channels);

Mat Y = channels[0]; // U = channels[1], V = channels[2]

float\*\* PDF\_RGB = cal\_PDF\_RGB(input); // PDF of Input image(RGB) : [L][3]

float\* CDF\_YUV = cal\_CDF(Y); // CDF of Y channel image

// histogram equalization on Y channel

hist\_eq(Y, channels[0], trans\_func\_eq\_YUV, CDF\_YUV);

// merge Y, U, V channels

merge(channels, 3, equalized\_YUV);

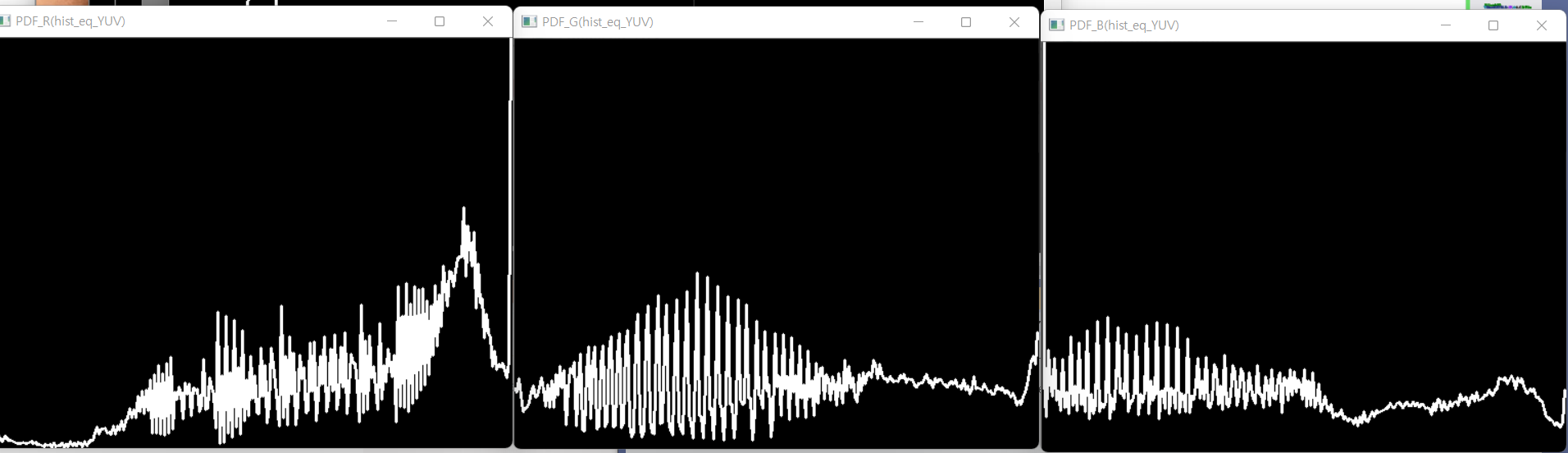
// YUV -> RGB (use "CV\_YUV2RGB" flag)

cvtColor(equalized\_YUV, equalized\_YUV, CV\_YUV2RGB);

**2. Results**

**포유류, 고양이, 보는, 실내이(가) 표시된 사진

자동 생성된 설명텍스트이(가) 표시된 사진

자동 생성된 설명**

**hist\_matching\_gray.cpp**

**1. Code Explanation**

1) void hist\_eq(Mat& input, G\* trans\_func, float\* CDF)

: histogram equalization(but only computing transfer function)

-input: input matrix

-trans\_func: s=trans\_func(r)=(L-1)CDF(r)

-CDF:Cumulative Distribution Function

2) void inverse\_function(G\* trans\_func, G\* trans\_func\_inv

: compute inverse function of trans\_func and save it in trans\_func\_inv

-trans\_func: input function

-trans\_func\_inv: inverse function

for (int i = 0; i < L; i++)

trans\_func\_inv[trans\_func[i]] = i;

for (int i = 0; i < L; i++) { //역함수 값이 빈 경우 가장 근처 값으로 채워준다

if (trans\_func2\_inv[i] == 0) {

int j = i;

while (trans\_func2\_inv[j] == 0 && j>=0)

j--;

trans\_func2\_inv[i] = trans\_func2\_inv[j];

}

}

3) void hist\_matching(Mat& input,Mat& ref, Mat& matched, G\* trans\_func\_ma, G\* trans\_func1, G\* trans\_func2, G\* trans\_func2\_inv, float\* CDF, float\* ref\_CDF)

: histogram matching function

-input:input image

-ref: reference image

-matched: matched image

-trans\_func\_ma: transfer function of histogram matching

-trans\_func1: s=T(r)-trans\_func2: s=G(z)

-trans\_func2\_inv: inverse of s=G(z)

-CDF: CDF of input image

-ref\_CDf: CDF of reference image

hist\_eq(input, trans\_func1, CDF); //compute s=T(r)

hist\_eq(ref, trans\_func2, ref\_CDF); //compute s=G(z)

inverse\_function(trans\_func2, trans\_func2\_inv); ////compute inverse function of s=G(z)

for (int i = 0;i < L;i++) { //intensity mapping from r to z -> z=G-1(T(r))

trans\_func\_ma[i] = trans\_func2\_inv[trans\_func1[i]];

}

// perform the transfer function -> matched image를 구한다

for (int i = 0; i < input.rows; i++)

for (int j = 0; j < input.cols; j++)

matched.at<G>(i, j) = trans\_func\_ma[input.at<G>(i, j)];

4) int main()

// transfer functions

G trans\_func\_ma[L] = { 0 };

G trans\_func1[L] = { 0 }; // s=T(r)

G trans\_func2[L] = { 0 }; // s=G(z

G trans\_func2\_inv[512 \* 512] = { 0 }; // inverse function of s=G(z)

//histogram matching on a grayscale image

hist\_matching(input\_gray, ref\_image\_gray, matched, trans\_func\_ma, trans\_func1, trans\_func2, trans\_func2\_inv, CDF, ref\_CDF);

//plot histogram(PDF)

int hist\_w = 512; int hist\_h = 400;

int bin\_w = cvRound((double)hist\_w / L);

Mat histImage\_PDF(hist\_h, hist\_w, CV\_8UC1, Scalar(0, 0, 0)); //PDF 나타낼 image 초기화

for (int i = 1;i < L;i++) { //line 함수를 이용해 이웃한 두 점을 연결하여 histogram을 그린다

line(histImage\_PDF, Point(bin\_w \* (i - 1), hist\_h - cvRound(PDF[i - 1] \* 16000)),

Point(bin\_w \* i, hist\_h - cvRound(PDF[i] \* 16000)), Scalar(255, 0, 0), 2, 8, 0);

} //그래프의 세로축을 0.025로 맞추기 위해 PDF에 400\*40을 곱한다

//plot histogram(matched\_PDF)

Mat histImage\_matched\_PDF(hist\_h, hist\_w, CV\_8UC1, Scalar(0, 0, 0)); //CDF 나타낼 image 초기화

for (int i = 1;i < L;i++) {

line(histImage\_matched\_PDF, Point(bin\_w \* (i - 1), hist\_h - cvRound(matched\_PDF\_gray[i - 1] \* 16000)),Point(bin\_w \* i, hist\_h - cvRound(matched\_PDF\_gray[i] \* 16000)), Scalar(255, 0, 0), 2, 8, 0);

}

//show histogram

namedWindow("PDF", WINDOW\_AUTOSIZE);

imshow("PDF", histImage\_PDF);

namedWindow("PDF(hist\_eq)", WINDOW\_AUTOSIZE);

imshow("PDF(hist\_eq)", histImage\_matched\_PDF);

**2. Results**

Reference image: lena.jpg

여자, 사람, 머리카락이(가) 표시된 사진

자동 생성된 설명

Input/output image

**고양이, 포유류, 여우, 실내이(가) 표시된 사진

자동 생성된 설명**

Histogram of original/output image

**텍스트, 음악이(가) 표시된 사진

자동 생성된 설명**

**hist\_matching\_YUV.cpp**

**1. Code Explanation**

1) void hist\_eq(Mat& input, G\* trans\_func, float\* CDF)

: histogram equalization(but only computing transfer function)

-input: input matrix

-trans\_func: s=trans\_func(r)=(L-1)CDF(r)

-CDF:Cumulative Distribution Function

2) void inverse\_function(G\* trans\_func, G\* trans\_func\_inv

: compute inverse function of trans\_func and save it in trans\_func\_inv

-trans\_func: input function

-trans\_func\_inv: inverse function

for (int i = 0; i < L; i++)

trans\_func\_inv[trans\_func[i]] = i;

for (int i = 0; i < L; i++) { //역함수 값이 빈 경우 가장 근처 값으로 채워준다

if (trans\_func2\_inv[i] == 0) {

int j = i;

while (trans\_func2\_inv[j] == 0 && j>=0)

j--;

trans\_func2\_inv[i] = trans\_func2\_inv[j];

}

}

3) void hist\_matching(Mat& input,Mat& ref, Mat& matched, G\* trans\_func\_ma, G\* trans\_func1, G\* trans\_func2, G\* trans\_func2\_inv, float\* CDF, float\* ref\_CDF)

: histogram matching function

-input:input image

-ref: reference image

-matched: matched image

-trans\_func\_ma: transfer function of histogram matching

-trans\_func1: s=T(r)-trans\_func2: s=G(z)

-trans\_func2\_inv: inverse of s=G(z)

-CDF: CDF of input image

-ref\_CDf: CDF of reference image

hist\_eq(input, trans\_func1, CDF); //compute s=T(r)

hist\_eq(ref, trans\_func2, ref\_CDF); //compute s=G(z)

inverse\_function(trans\_func2, trans\_func2\_inv); ////compute inverse function of s=G(z)

for (int i = 0;i < L;i++) { //intensity mapping from r to z -> z=G-1(T(r))

trans\_func\_ma[i] = trans\_func2\_inv[trans\_func1[i]];

}

// perform the transfer function -> matched image를 구한다

for (int i = 0; i < input.rows; i++)

for (int j = 0; j < input.cols; j++)

matched.at<G>(i, j) = trans\_func\_ma[input.at<G>(i, j)];

4) int main()

// transfer functions

G trans\_func\_ma[L] = { 0 };

G trans\_func1[L] = { 0 };

G trans\_func2[L] = { 0 };

//histogram matching on a grayscale image

hist\_matching(Y, ref\_Y, channels[0], trans\_func\_ma, trans\_func1, trans\_func2, trans\_func2\_inv, CDF\_YUV, ref\_CDF\_YUV);

// merge Y, U, V channels

merge(channels, 3, matched\_YUV);

// YUV -> RGB (use "CV\_YUV2RGB" flag)

cvtColor(matched\_YUV, matched\_YUV, CV\_YUV2RGB);

// equalized PDF (YUV)

float\*\* matched\_PDF\_YUV = cal\_PDF\_RGB(matched\_YUV);

//plot histogram(PDF)

//plot histogram(matched\_PDF\_YUV

**2. Results**

Reference image: lena.jpg

여자, 사람, 머리카락이(가) 표시된 사진

자동 생성된 설명

Input/output image

**포유류, 고양이, 실내, 여우이(가) 표시된 사진

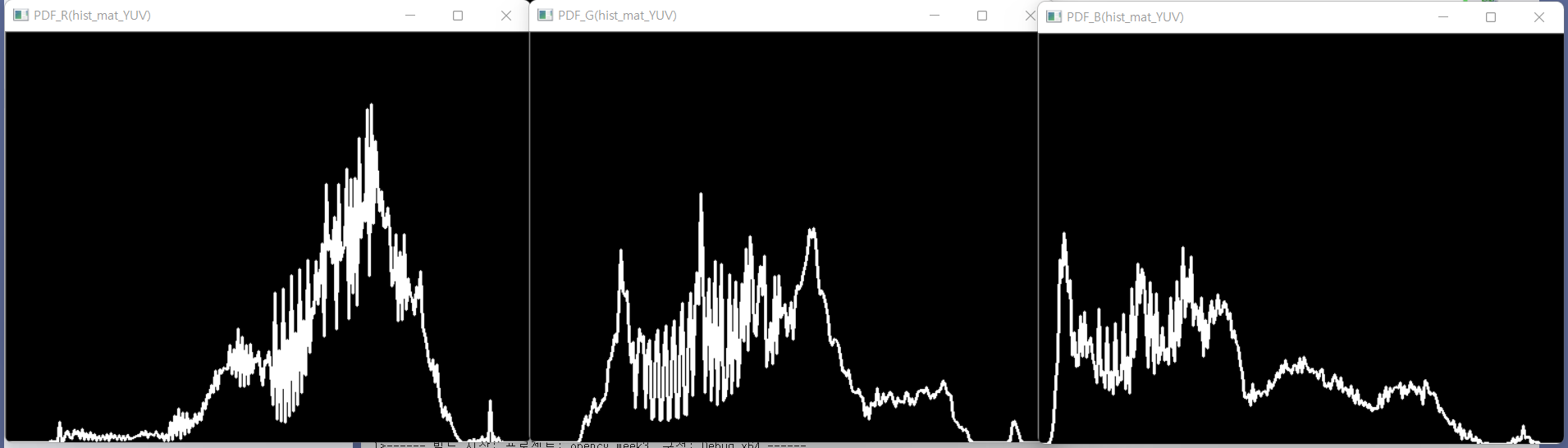
자동 생성된 설명**

Histogram of the original image(RGB)

**텍스트이(가) 표시된 사진

자동 생성된 설명**

Histogram of the output image(RGB)

****