영상처리 실제 - 5주차 실습

2023254009 최현동

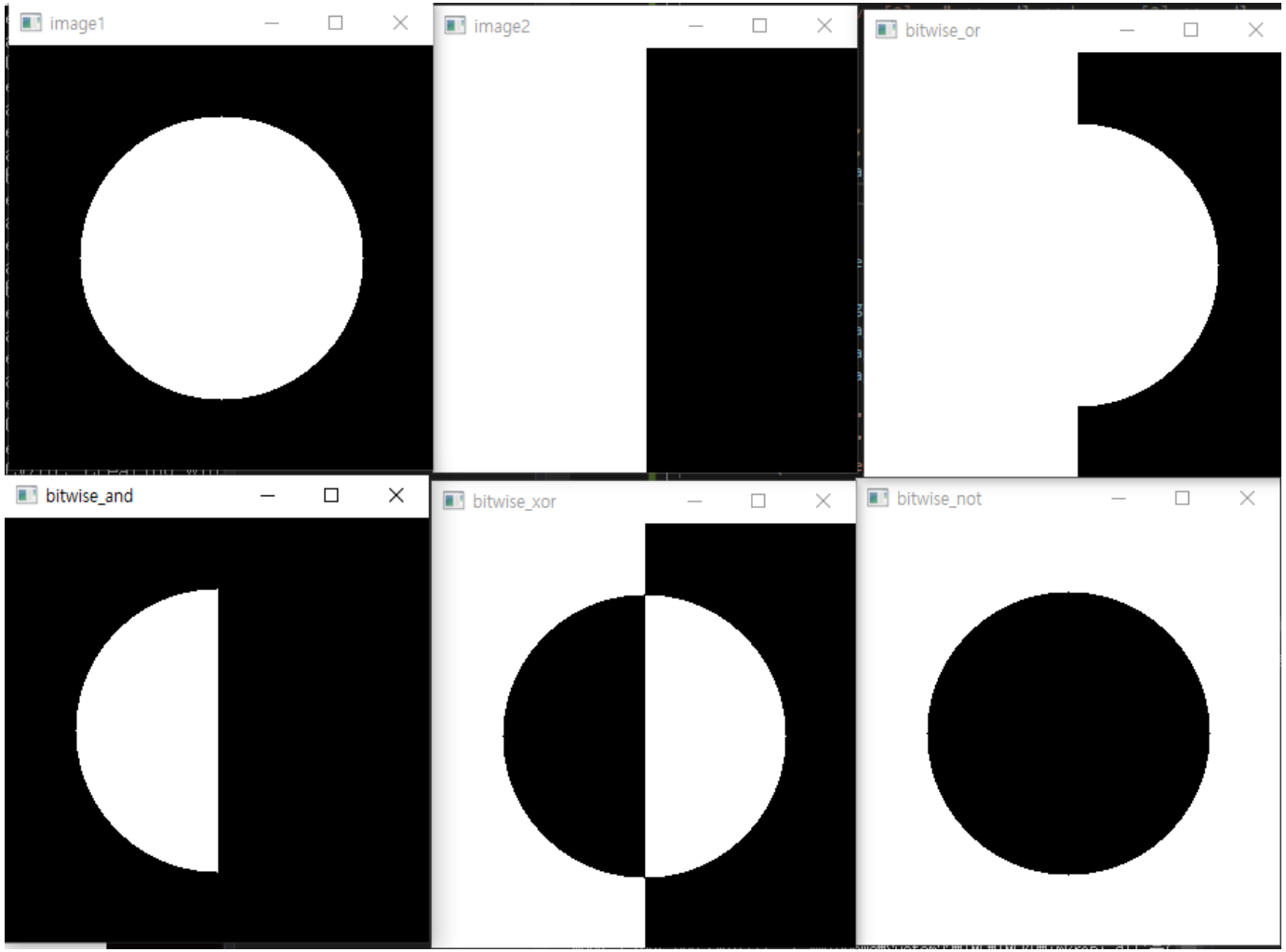
(5) – p.8

```
    //5 - p8
#if 1
    Mat ch0(3, 4, CV_8U, Scalar(10));
    Mat ch1(3, 4, CV_8U, Scalar(20));
    Mat ch2(3, 4, CV_8U, Scalar(30));

    Mat bgr_arr[] = { ch0, ch1, ch2 };
    Mat bgr;
    merge(bgr_arr, 3, bgr);
    vector<Mat> bgr_vec;
    split(bgr, bgr_vec);

    cout << "[ch0] = " << endl << ch0 << endl;
    cout << "[ch1] = " << endl << ch1 << endl;
    cout << "[ch2] = " << endl << ch2 << endl << endl;

    cout << "[bgr] = " << endl << bgr << endl << endl;
    cout << "[bgr_vec[0] = " << endl << bgr_vec[0] << endl;
    cout << "[bgr_vec[1] = " << endl << bgr_vec[1] << endl;
    cout << "[bgr_vec[2] = " << endl << bgr_vec[2] << endl;

#endif
```

```
[ch0] =
 10,  10,  10,  10;
 10,  10,  10,  10;
 10,  10,  10,  10]
[ch1] =
 20,  20,  20,  20;
 20,  20,  20,  20;
 20,  20,  20,  20]
[ch2] =
 30,  30,  30,  30;
 30,  30,  30,  30;
 30,  30,  30,  30]

[bgr] =
 10,  20,  30,  10,  20,  30,  10,  20,  30,  10,  20,  30;
 10,  20,  30,  10,  20,  30,  10,  20,  30,  10,  20,  30;
 10,  20,  30,  10,  20,  30,  10,  20,  30,  10,  20,  30]

[bgr_vec[0] =
 10,  10,  10,  10;
 10,  10,  10,  10;
 10,  10,  10,  10]
[bgr_vec[1] =
 20,  20,  20,  20;
 20,  20,  20,  20;
 20,  20,  20,  20]
[bgr_vec[2] =
 30,  30,  30,  30;
 30,  30,  30,  30;
 30,  30,  30,  30]
```

(5) – p.16

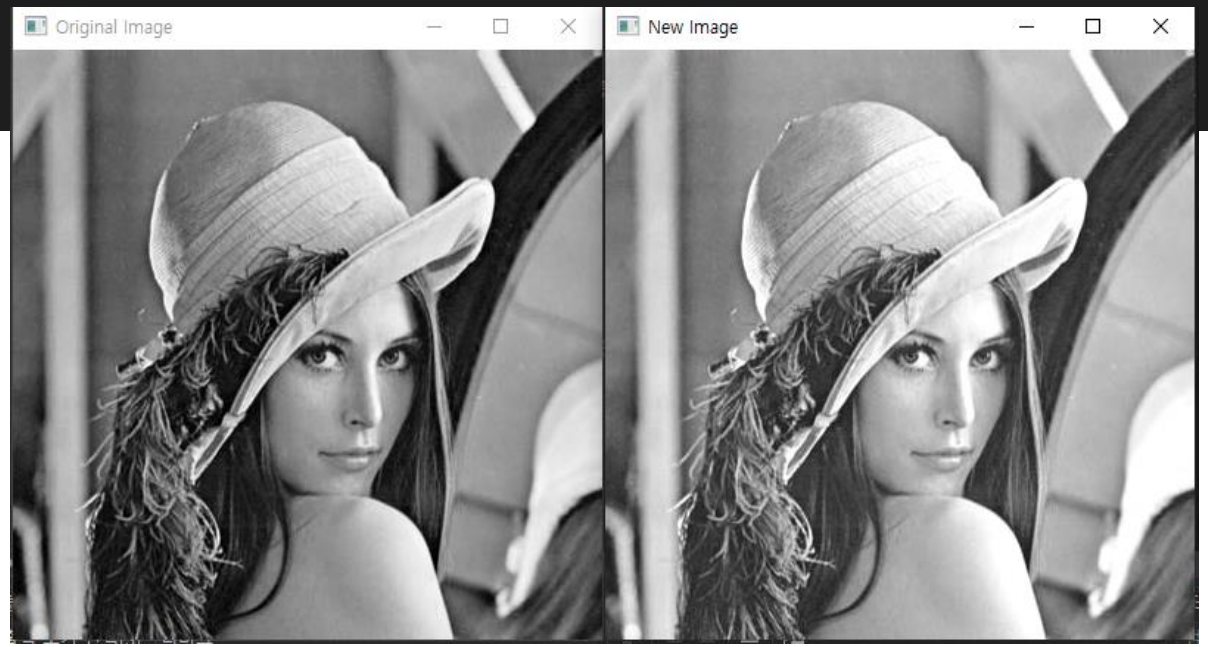
```
    //5 - p16
#if 1
    Mat image1(300, 300, CV_8U, Scalar(0));
    Mat image2(300, 300, CV_8U, Scalar(0));
    Mat image3, image4, image5, image6;

    Point center = image1.size() / 2;
    circle(image1, center, 100, Scalar(255), -1);
    rectangle(image2, Point(0, 0), Point(150, 300), Scalar(255), -1);

    bitwise_or(image1, image2, image3);
    bitwise_and(image1, image2, image4);
    bitwise_xor(image1, image2, image5);
    bitwise_not(image1, image6);

    imshow("image1", image1);
    imshow("image2", image2);
    imshow("bitwise_or", image3);
    imshow("bitwise_and", image4);
    imshow("bitwise_xor", image5);
    imshow("bitwise_not", image6);

    waitKey(0);
#endif
```



: 화소처리 – p.8

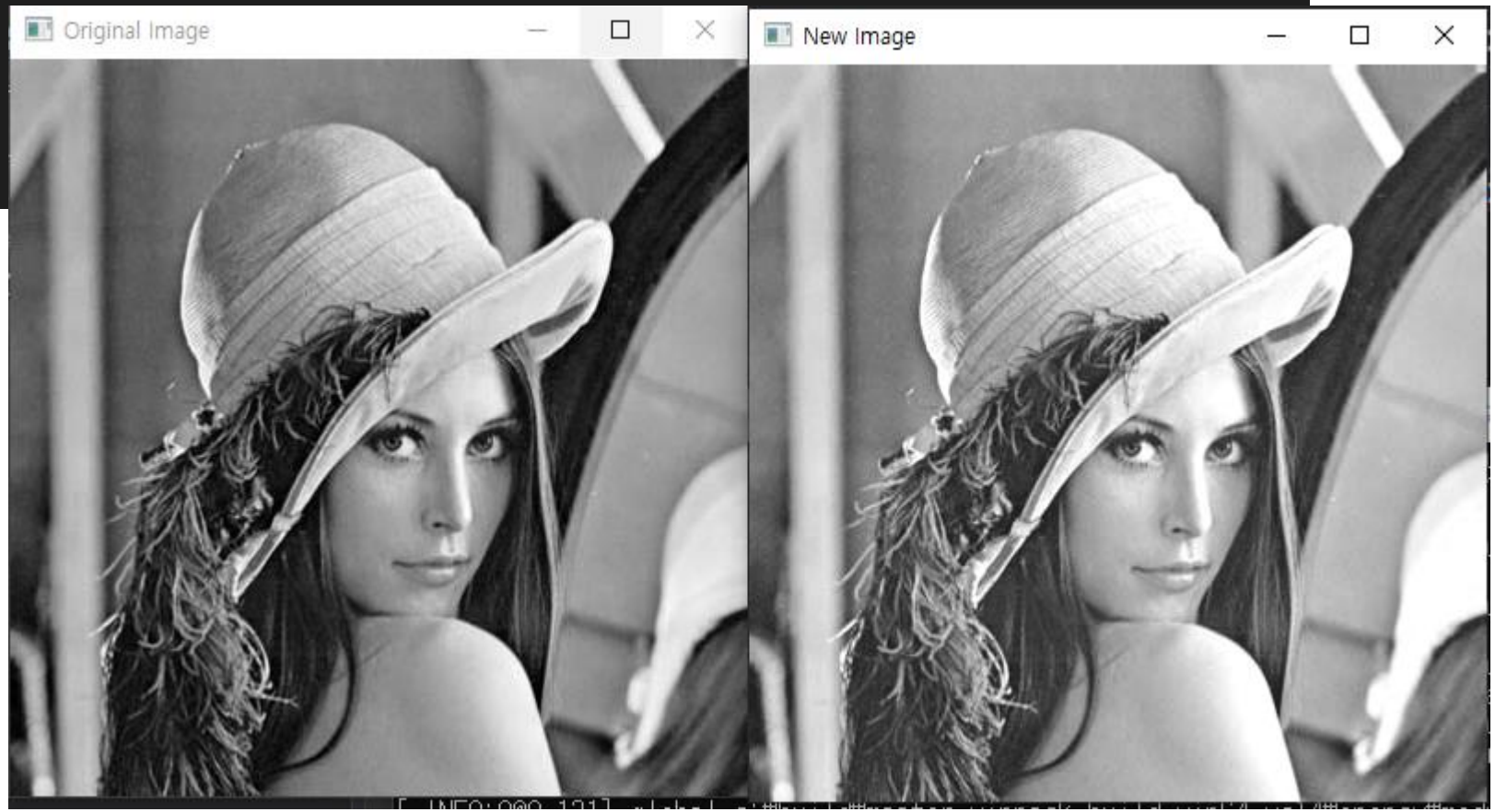
```
void brighten(Mat& img, int value)
{
    for (int r = 0; r < img.rows; r++)
    {
        for (int c = 0; c < img.cols; c++)
        {
            img.at<uchar>(r, c) = saturate_cast<uchar>(img.at<uchar>(r, c) + value);
        }
    }
}
```

```
    //화소처리 - p.8
#if 1
    Mat img = imread("D:\\1.개인폴더\\2.산업인공지능학과\\2.23년2학기(석사2학기)\\2.영상처리실제\\3.실습\\4.5주차실습\\3.Image\\lenna.jpg", IMREAD_GRAYSCALE);
    imshow("Original Image", img);

    brighten(img, 30);
    imshow("New Image", img);
    waitKey(0);
#endif
```



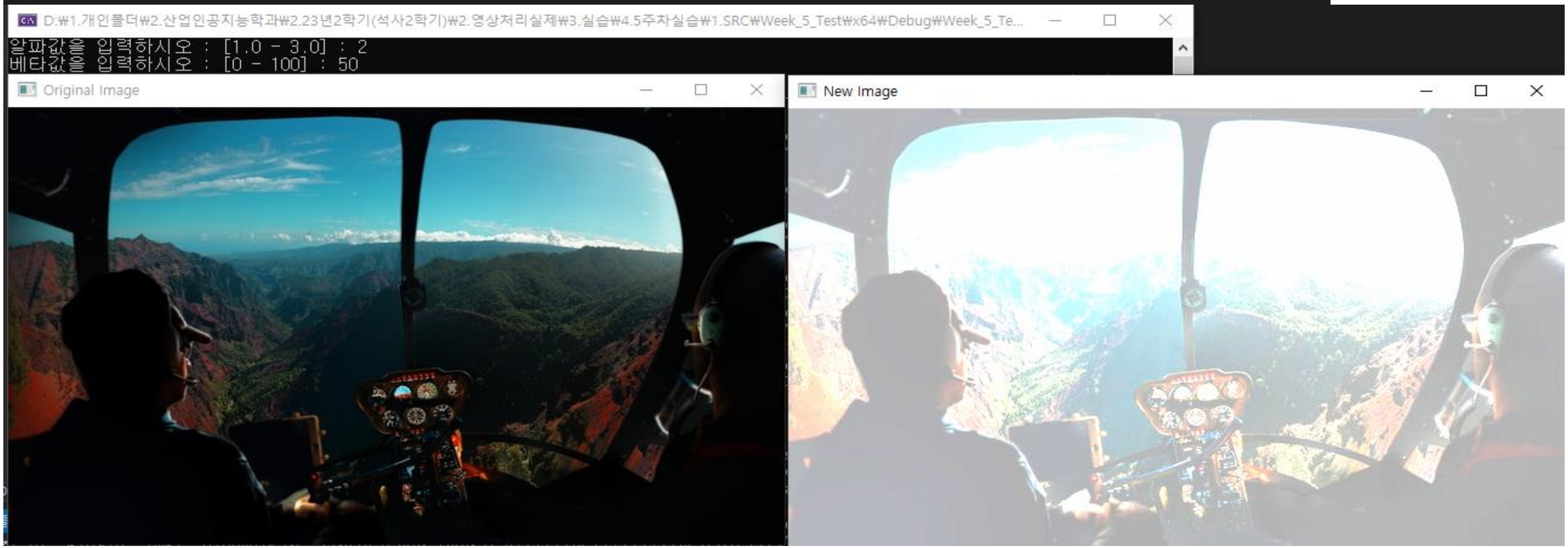
: 화소처리 – p.10

```cpp
//화소처리 - p.10
#if 1
Mat img = imread("D:\\1.개인폴더\\2.산업인공지능학과\\2.23년2학기(석사2학기)\\2.영상처리실제\\3.실습\\4.5주차실습\\3.Image\\lenna.jpg", IMREAD_GRAYSCALE);
imshow("Original Image", img);

for (int r = 0; r < img.rows; r++)
{
    uchar* p = img.ptr<uchar>(r);

    for (int c = 0; c < img.cols; ++c)
    {
        p[c] = saturate_cast<uchar> (p[c] + 30);
    }
}
imshow("New Image", img);

waitKey(0);
#endif
```



: 화소처리 – p.14

```cpp
    //화소처리 - p.14
#if 1
    double alpha = 1.0;
    int beta = 0;
    Mat image = imread("D:\\1.개인폴더\\2.산업인공지능학과\\2.23년2학기(석사2학기)\\2.영상처리실제\\3.실습\\4.5주차실습\\3.Image\\contrast.jpg");

    Mat oimage;
    cout << "알파값을 입력하시오 : [1.0 - 3.0] : "; cin >> alpha;
    cout << "베타값을 입력하시오 : [0 - 100] : "; cin >> beta;
    image.convertTo(oimage, 1, alpha, beta);
    imshow("Original Image", image);
    imshow("New Image", oimage);
    waitKey(0);
#endif
```

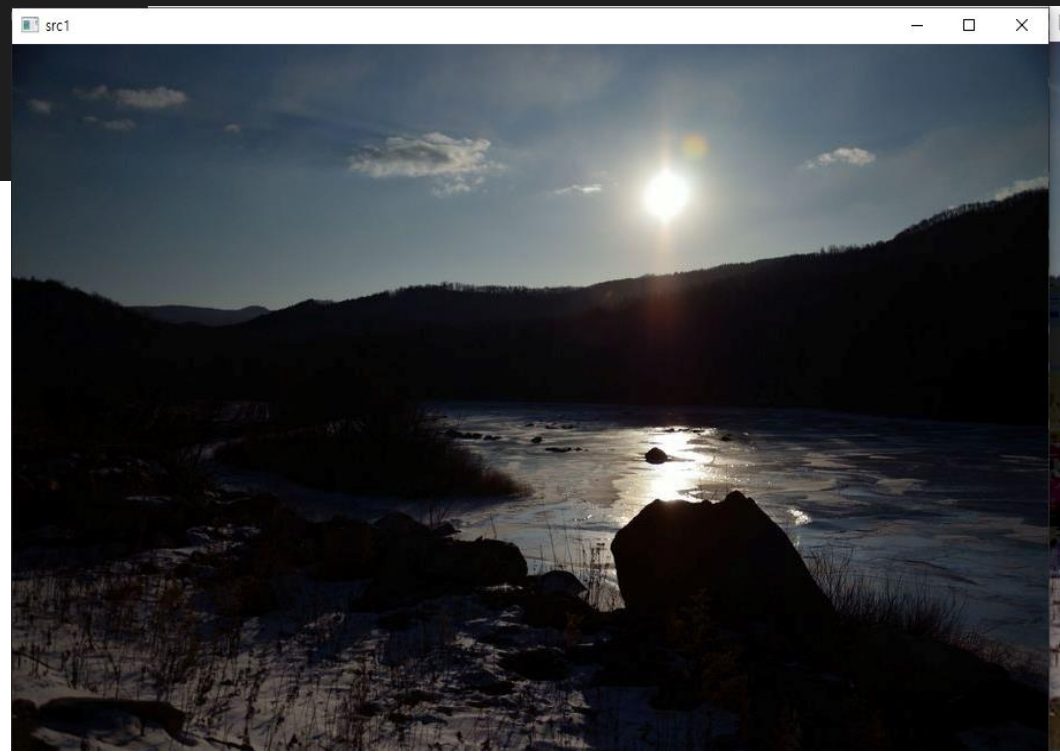


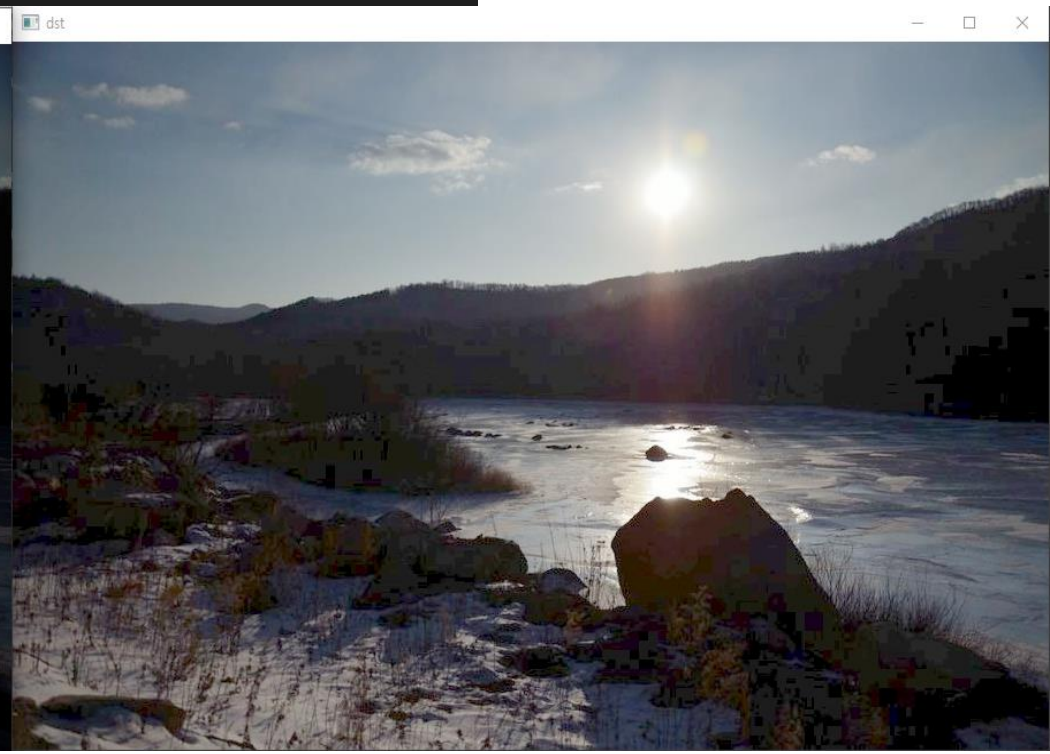
: 화소처리 – p.21

```
    //화소처리 - p.21
#if 1
    src = imread("D:\\1.개인폴더\\2.산업인공지능학과\\2.23년2학기(석사2학기)\\2.영상처리실제\\3.실습\\4.5주차실습\\3.Image\\lenna.jpg");
    cvtColor(src, src_gray, COLOR_BGR2GRAY);
    namedWindow("결과영상", WINDOW_AUTOSIZE);
    createTrackbar("임계값", "결과영상", &threshold_value, 255, Threshold_Demo);

    Threshold_Demo(0, 0);
    while (true)
    {
        int c;
        c = waitKey(20);
        if ((char)c == 27)
        {
            break;
        }
    }
#endif

Mat src, src_gray, dst;
int threshold_value = 0;
int threshold_type = 0;

void Threshold_Demo(int, void*)
{
    threshold(src_gray, dst, threshold_value, 255, threshold_type);
    imshow("결과영상", dst);
}
```

: 화소처리 – p.31

```cpp
    //화소처리 - p.31
#if 1
    Mat src1, src2, dst;
    double gamma = 0.5;
    src1 = imread("D:\\1.개인폴더\\2.산업인공지능학과\\2.23년2학기(석사2학기)\\2.영상처리실제\\3.실습\\4.5주차실습\\3.Image\\gamma1.jpg");
    if (src1.empty())
    {
        cout << "영상을 읽을수 없습니다." << endl;
        return -1;
    }
    Mat table(1, 256, CV_8U);
    uchar* p = table.ptr();
    for (int i = 0; i < 256; i++)
    {
        p[i] = saturate_cast<uchar>(pow(i / 255.0, gamma) * 255.0);
    }

    LUT(src1, table, dst);
    imshow("src1", src1);
    imshow("dst", dst);
    waitKey(0);
#endif
```



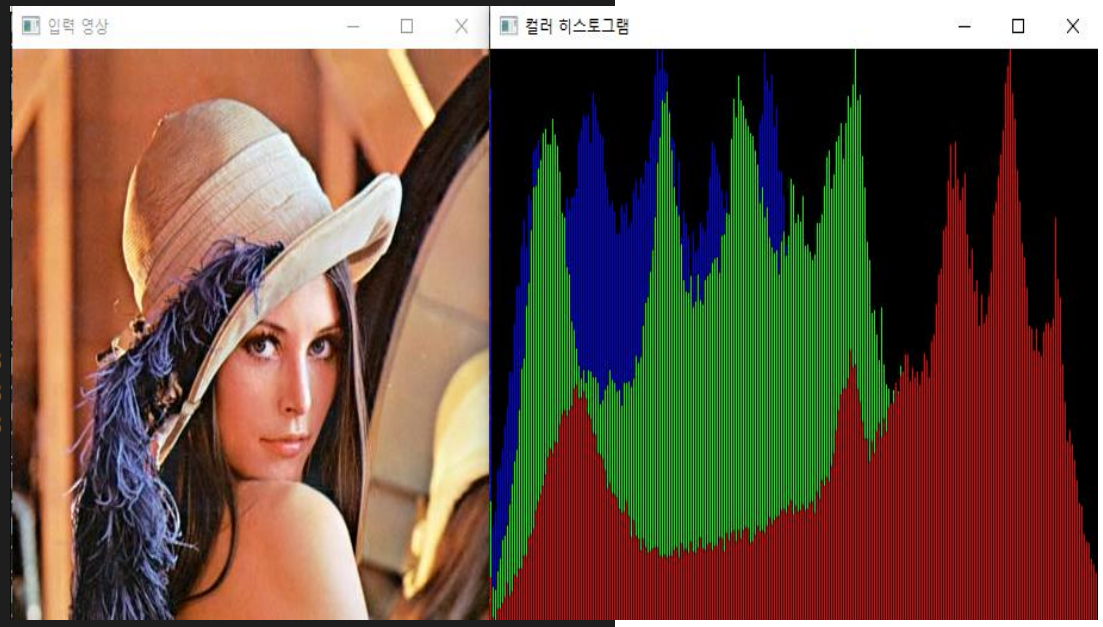


: 히스토그램 – p.11

```
void drawHist(int histogram[])
{
    int hist_w = 512; //히스토그램 영상의 폭
    int hist_h = 400; //히스토그램 영상의 높이
    int bin_w = cvRound((double)hist_w / 256); //빈의 폭

    //히스토그램이 그려지는 영상(칼라로 정의)
    Mat histImage(hist_h, hist_w, CV_8UC3, Scalar(255, 255, 255));

    //히스토그램의 최대값을 찾는다.
    int max = histogram[0];
    for (int i = 1; i < 256; i++)
    {
        if (max < histogram[i])
        {
            max = histogram[i];
        }
    }

    //히스토그램 배열을 최대값으로 정규화 한다.(최대값이 최대높이가 되도록)
    for (int i = 0; i < 255; i++)
    {
        histogram[i] = floor(((double)histogram[i] / max) * histImage.rows);
    }

    //히스토그램의 값을 빨강색 막대로 그린다.
    for (int i = 0; i < 255; i++)
    {
        line(histImage, Point(bin_w * (i), hist_h), Point(bin_w * (i), hist_h - histogram[i]), Scalar(0, 0, 255));
    }

    imshow("Histogram", histImage);
}

    //히스토그램 - p.11
#if 1
    Mat src = imread("D:\\1.개인폴더\\2.산업인공지능학과\\2.23년2학기(석사2학기)\\2.영상처리실제\\3.실습\\4.5주차실습\\3.Image\\lenna.jpg", IMREAD_GRAYSCALE);
    imshow("Input Image", src);
    int histogram[256] = { 0 };

    for (int y = 0; y < src.rows; y++)
    {
        for (int x = 0; x < src.cols; x++)
        {
            histogram[(int)src.at<uchar>(y, x)]++;
        }
    }

    drawHist(histogram);
    waitKey(0);
#endif
```

: 히스토그램 – p.15

```cpp
//히스토그램 - p.15
#if 1
Mat src = imread("D:\\1.개인폴더\\2.산업인공지능학과\\2.23년2학기(석사2학기)\\2.영상처리실제\\3.실습\\4.5주차실습\\3.Image\\lenna.jpg");
if (src.empty())
{
    return -1;
}
vector<Mat> bgr_planes;
split(src, bgr_planes);
int histSize = 256;
float range[] = { 0, 256 };
const float* histRange = { range };
bool uniform = true, accumlate = false;

Mat b_hist, g_hist, r_hist;
calcHist(&bgr_planes[0], 1, 0, Mat(), b_hist, 1, &histSize, &histRange, uniform, accumlate);
calcHist(&bgr_planes[1], 1, 0, Mat(), g_hist, 1, &histSize, &histRange, uniform, accumlate);
calcHist(&bgr_planes[2], 1, 0, Mat(), r_hist, 1, &histSize, &histRange, uniform, accumlate);
//막대그래프가 그려지는 영상을 생성한다.
int hist_w = 512, hist_h = 400;
int bin_w = cvRound((double)hist_w / histSize);// 상자의 폭
Mat histImage(hist_h, hist_w, CV_8UC3, Scalar(0, 0, 0));
//값들이 영상을 벗어나지 않도록 정규화한다.
normalize(b_hist, b_hist, 0, histImage.rows, NORM_MINMAX, -1, Mat());
normalize(g_hist, g_hist, 0, histImage.rows, NORM_MINMAX, -1, Mat());
normalize(r_hist, r_hist, 0, histImage.rows, NORM_MINMAX, -1, Mat());

//  히스토그램의 값을 막대로 그린다.
for (int i = 0; i < 255; i++)
{
    line(histImage, Point(bin_w * (i), hist_h), Point(bin_w * (i), hist_h - b_hist.at<float>(i)), Scalar(255, 0, 0));
    line(histImage, Point(bin_w * (i), hist_h), Point(bin_w * (i), hist_h - g_hist.at<float>(i)), Scalar(0, 255, 0));
    line(histImage, Point(bin_w * (i), hist_h), Point(bin_w * (i), hist_h - r_hist.at<float>(i)), Scalar(0, 0, 255));
}

imshow("입력 영상", src);
imshow("컬러 히스토그램", histImage);
waitKey();
#endif
```



: 히스토그램 – p.19

```
int stretch(int x, int r1, int s1, int r2, int s2)
{
    float result;
    if (0 <= x && x <= r1)
    {
        result = s1 / r1 * x;
    }
    else if (r1 < x && x <= r2)
    {
        result = ((s2 - s1) / (r2 - r1)) * (x - r1) + s1;
    }
    else if (r2 < x && x <= 255)
    {
        result = ((255 - s2) / (255 - r2)) * (x - r2) + s2;
    }
    return (int)result;
}
```


```
    //히스토그램 - p.19
#if 1
    Mat image = imread("D:\\1.개인폴더\\2.산업인공지능학과\\2.23년2학기(석사2학기)\\2.영상처리실제\\3.실습\\4.5주차실습\\3.Image\\crayfish.jpg");
    Mat new_image = image.clone();

    int r1, r2, s1, s2;
    cout << "r1을 입력 하시오 : "; cin >> r1;
    cout << "r2을 입력 하시오 : "; cin >> r2;
    cout << "s1을 입력 하시오 : "; cin >> s1;
    cout << "s2을 입력 하시오 : "; cin >> s2;

    for (int y = 0; y < image.rows; y++)
    {
        for (int x = 0; x < image.cols; x++)
        {
            for (int c = 0; c < 3; c++)
            {
                int output = stretch(image.at<Vec3b>(y, x)[c], r1, s1, r2, s2);
                new_image.at<Vec3b>(y, x)[c] = saturate_cast<uchar>(output);
            }
        }
    }

    imshow("입력영상", image);
    imshow("출력영상", new_image);
    waitKey();
#endif
```

: 히스토그램 – p.24 - 1

```cpp
void calc_Histo(const Mat& image, Mat& hist, int bins, int range_max = 256)
{
    int histSize[] = { bins };
    float range[] = { 0, (float)range_max };
    int channels[] = { 0 };
    const float* ranges[] = { range };

    calcHist(&image, 1, channels, Mat(), hist, 1, histSize, ranges);
}

void draw_Histo(Mat hist, Mat &hist_img, Size size = Size(256, 200))
{
    hist_img = Mat(size, CV_8U, Scalar(255));
    float bin = (float)hist_img.cols / hist.rows;
    normalize(hist, hist, 0, hist_img.rows, NORM_MINMAX);

    for (int i = 0; i < hist.rows; i++)
    {
        float start_x = i * bin;
        float end_x = (i + 1) * bin;
        Point2f pt1(start_x, 0);
        Point2f pt2(end_x, hist.at<float>(i));

        if(pt2.y > 0)
        {
            rectangle(hist_img, pt1, pt2, Scalar(0), -1);
        }
    }
    flip(hist_img, hist_img, 0);
}

void create_hist(Mat img, Mat& hist, Mat& hist_img)
{
    int histsize = 256, range = 256;
    calc_Histo(img, hist, histsize, range);
    draw_Histo(hist, hist_img);
}
```

```cpp
//히스토그램 - p.24
#if 1
    Mat image = imread("D:\\1.개인폴더\\2.산업인공지능학과\\2.23년2학기(석사2학기)\\2.영상처리실제\\3.실습\\4.5주차실습\\3.Image\\equalize_test.jpg", 0);
    CV_Assert(!image.empty());

    Mat hist, dst1, dst2, hist_img, hist_img1, hist_img2;
    create_hist(image, hist, hist_img);

    //히스토그램 누적합 계산
    Mat accum_hist = Mat(hist.size(), hist.type(), Scalar(0));
    accum_hist.at<float>(0) = hist.at<float>(0);
    for (int i = 1; i < hist.rows; i++)
    {
        accum_hist.at<float>(i) = accum_hist.at<float>(i - 1) + hist.at<float>(i);
    }

    accum_hist /= sum(hist)[0];
    accum_hist *= 255;
    dst1 = Mat(image.size(), CV_8U);
    for (int i = 0; i < image.rows; i++)
    {
        for (int j = 0; j < image.cols; j++)
        {
            int idx = image.at<uchar>(i, j);
            dst1.at<uchar>(i, j) = (uchar)accum_hist.at<float>(idx);
        }
    }

    equalizeHist(image, dst2);
    create_hist(dst1, hist, hist_img1);
    create_hist(dst2, hist, hist_img2);

    imshow("image", image); imshow("img_hist", hist_img);
    imshow("dst1-User", dst1);  imshow("User-hist", hist_img1);
    imshow("dst2-OpenCV", dst2);    imshow("OpenCV_hist", hist_img2);
    waitKey();

#endif
```

: 히스토그램 – p.24 - 2

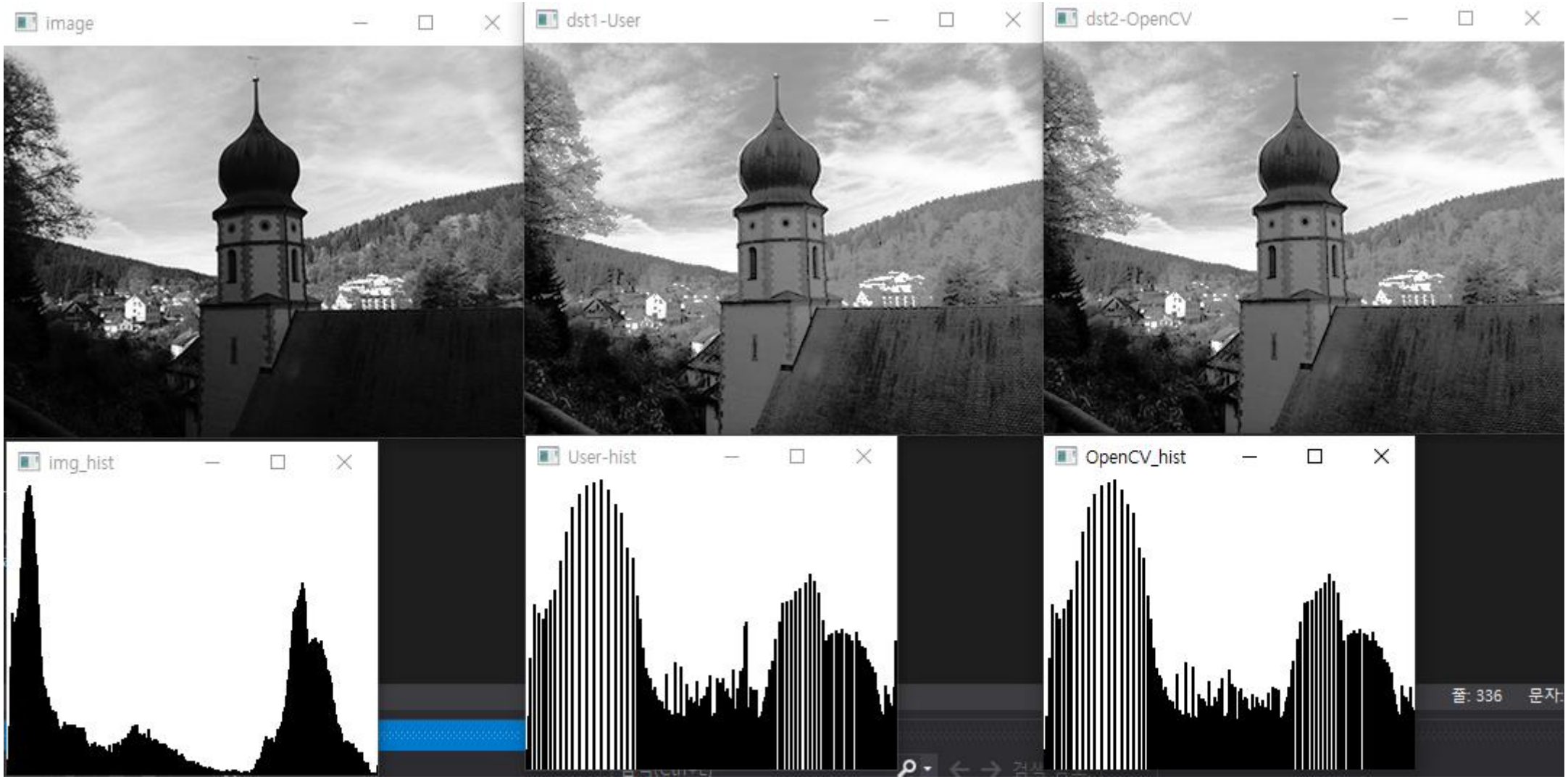


image
dst1-User
dst2-OpenCV
img_hist
User-hist
OpenCV_hist
줄: 336
문자: