2021-1 DLIP LAB #1

Temperature Measurement

21500264 Seongryeong Park
21600372 Yoonkyoung Song

1. Introduction

This experiment deals with various color image segmentation methods including filtering, InRange function, thresholding, morphology, and finding contours to identify the range of exposed skin (especially face) from the video of IR images of several people. To measure the face temperature, raw video should be analyzed to get a HSV value of each person. In addition, HSV should be mapped intensity value of the gray scale video to real temperature value between 25℃ to 40℃. Then, the maximum and average temperature must be displayed on the video and a warning sign if the average temperature is above 38℃. The appropriate standards of the mapping function which convert temperature from intensity will be mainly concerned, as well as type of morphology and contouring procedure.

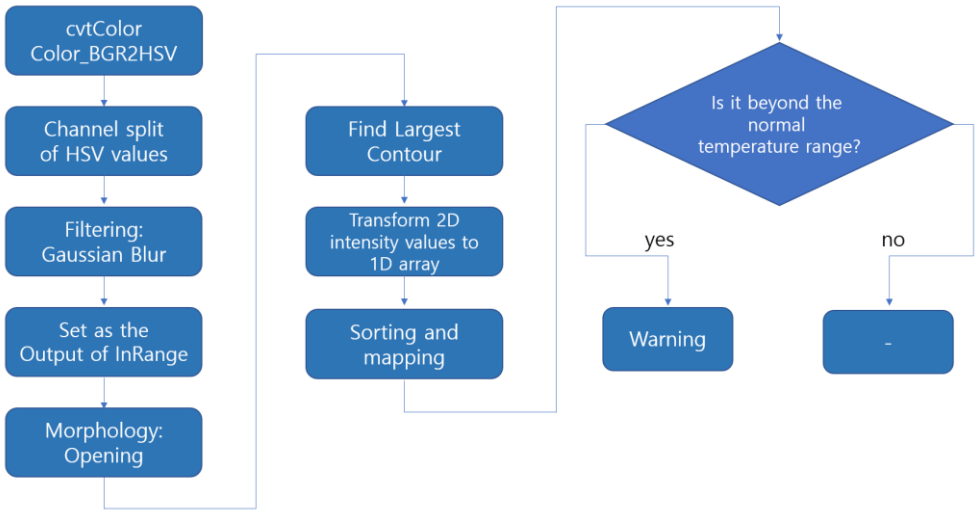


Figure 1. Flow Diagram

2. Procedure

2.1 Face Segmentation

2.1.1 Raw Image Processing

Color models that can be used to maximize the performance of image processing algorithms include

RGB, HSV, and grayscale. In the RGB model, the black color is expressed as (R,G,B) = (0,0,0) and the white as (R,G,B) = (255,255,255). Thus, using this model, we express colors in combinations of R, G, and B colors. On the other hand, HSV models express color in three components: hue, saturation, and brightness(value). HSV models can easily classify colors than RGB models because Hue has pure color information with a certain range. Hence, HSV is known as the most intuitive model to express color. Therefore, we proceed with the experiment procedure by converting images represented by RGB models into HSV models.

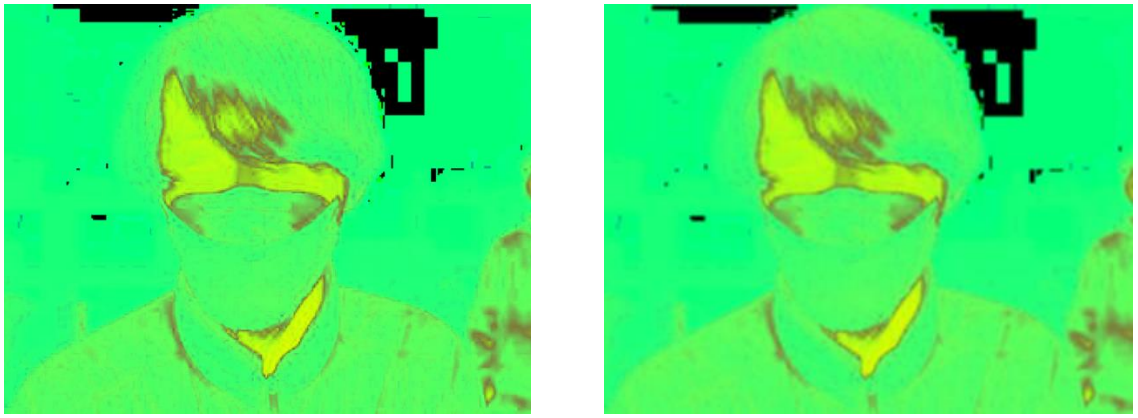
a) RGB Image **b) HSV Image**

Figure 2. Images before and after applying BGR2HSV function.

2.1.2 Filtering: Gaussian Blur

Gaussian Blur is a technique for blurring images using Gaussian Function. The surrounding pixel set and convolution for each pixel is performed via a mask calculated from the Gaussian Function. In the case of this experiment, we apply this effect because the image can reduce the precision of the contour due to noise. The blurring task utilizes the conversion with a low-pass kernel filter, and the noise and line existing in the high frequency range become blurry. Before applying the Gaussian-blur effect, it must be recognized that the higher the intensity of blurring, the less the sharpness. Unlike the average blur method, in which all pixels are weighted the same, it applies the highest weight to the pixels in the center. Therefore, selecting the appropriate kernel-size is significant because the Gaussian blur function can distort not only the noise but also the target outlines.

In addition to Gaussian filters, we have also considered the application of average filters etc. However, the difference in performance between filters was not significant because the source video was not a high definition. In addition, this effect was finally used, because, given that the Gaussian filter generally showed a clearer view of the target object than the average filter.



a) before Gaussian blurring **b) after Gaussian blurring**

Figure 3. Images before and after Gaussian blurring

2.1.3 In-range Function

We used the mouse to assign each person's skin(face) area. The HSV values were analyzed using the mouse interaction toolbox. Target area was dragged with a mouse to make an outline box and received the HSV value inside it. Thus, we could establish that every minimum and maximum HSV values. These values will be an important reference value for contour drawing.

$$(h_{min} = 0, \quad h_{max} = 50, \quad s_{min} = 180, \quad s_{max} = 255, \quad v_{min} = 80, \quad v_{max} = 215)$$

a) source image **b) image after InRange**

Figure 4. Identifying Region of Interest.

2.1.4 Morphology

Morphology refers to morphological operations used in image processing, such as noise removal, filling holes, and attaching broken lines. Morphology can only be applied to binary images(grayscale)

consisting of one channel value expressed in black and white. Morphology operations include erosion, dilation, opening, and closing. Erosion has the advantage of removing noise from dark areas, and dilation has the advantage of removing noise from bright areas. However, if we use these methods individually, it will inevitably result in deformation which causes the original shape to become thinner or thicker. Therefore, by using a combination of erosion and dilation, noise can be eliminated while maintaining the original shape.

In this experiment, we used an opening morphology transformation. Opening is the operation of the dilation after the erosion. Opening is effective in eliminating brighter noise than surroundings. It is also effective in separating shapes into individual objects. In this experiment, the average temperature and maximum temperature should be calculated by mapping the intensity value (0-255) of the gray scale to the measurement temperature range (25~40℃). Therefore, by eliminating brighter noise than the background, we can measure it more precisely at higher temperatures.

a) before morphology **b) after morphology**

Figure 5. Images using opening morphology.

2.1.5 Drawing Contours

Contour means a line that connects the same color value. Hence, the contouring image is connecting the edge boundary of the part with the same color intensity. We proceed the contouring with images that went through filtering, InRange function and morphology. Binary images should be used to get more precise image contour. We have already done the necessary task to do the contour through the process already described.

After finding all the contours, we will use only the contours that have the largest area among all contours. Since temperature is measured only with the face of a person close to the camera, a small area of contouring is not necessary. Hence, we saved only the value of the largest contour by declaring the value of the maximum area size 'maxArea' and its index 'largestComp'. Then, we can

contour only the area of the face of the person near the IR camera through this process. Furthermore, we used the BoundingRect function to draw a box on the edge of the contour area.

2.2 Temperature Measurement

2.2.1 Find Intensity form HSV

In a bid to map the intensity to the real temperature value (25~40℃) from the source video, we proceeded to the BGR2HSV function. Because we must obtain the value(brightness) means the 1-channel intensity value of the HSV model. Then, we converted the V(brightness) value to grayscale. In addition, we declared the variable 'mask' to receive the largest contour index and fill its areas with white color. In addition, we used bitwise_and() function to leave only the gray scale image portion of the contour area by combining the gray scale image and the contoured section. At this time, declare 'if' statements that limit the range and size of the boundingRect's x-coordinates drawn during the Contour process, so that no contour is left other than the person to measure the temperature.

a) mask b) V channel c) bitwise

Figure 5. Mask on V-channel image

2.2.2 Intensity Mapping

We are ready to map the intensity values to the temperature values by changing the 2D contour area calculated through the boundingRect function to 1D (descending). The value of V is the same as that of the IR camera, indicating the temperature according to the intensity, thus the value converting process can proceed. Intensity has values from 0 to 255, which are mapped to temperatures of 25℃ to 34℃. However, this relationship does not have linear properties, so it is mapped in a quadratic equation to make it sensitive in the temperature-specific interval of the personal temperature. We set the coefficients through a quadratic equation that passes three points $(0, 25)$, $(255, 40)$, and $(180, 36)$.

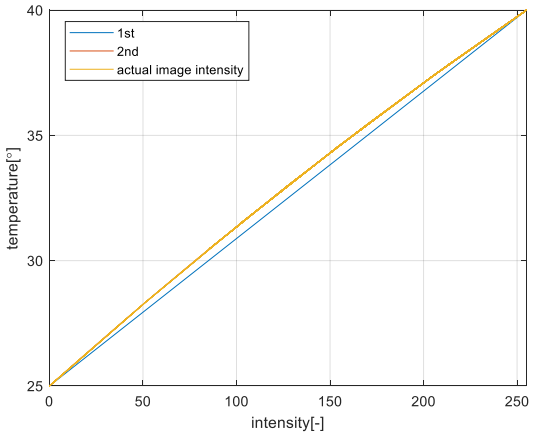


Figure 6. Intensity-Temperature Mapping

$$y_t = 4.943047496238989e^{-5} * x_i^2 + 0.052213625617881 * x_i + 25$$

2.2.3 Results

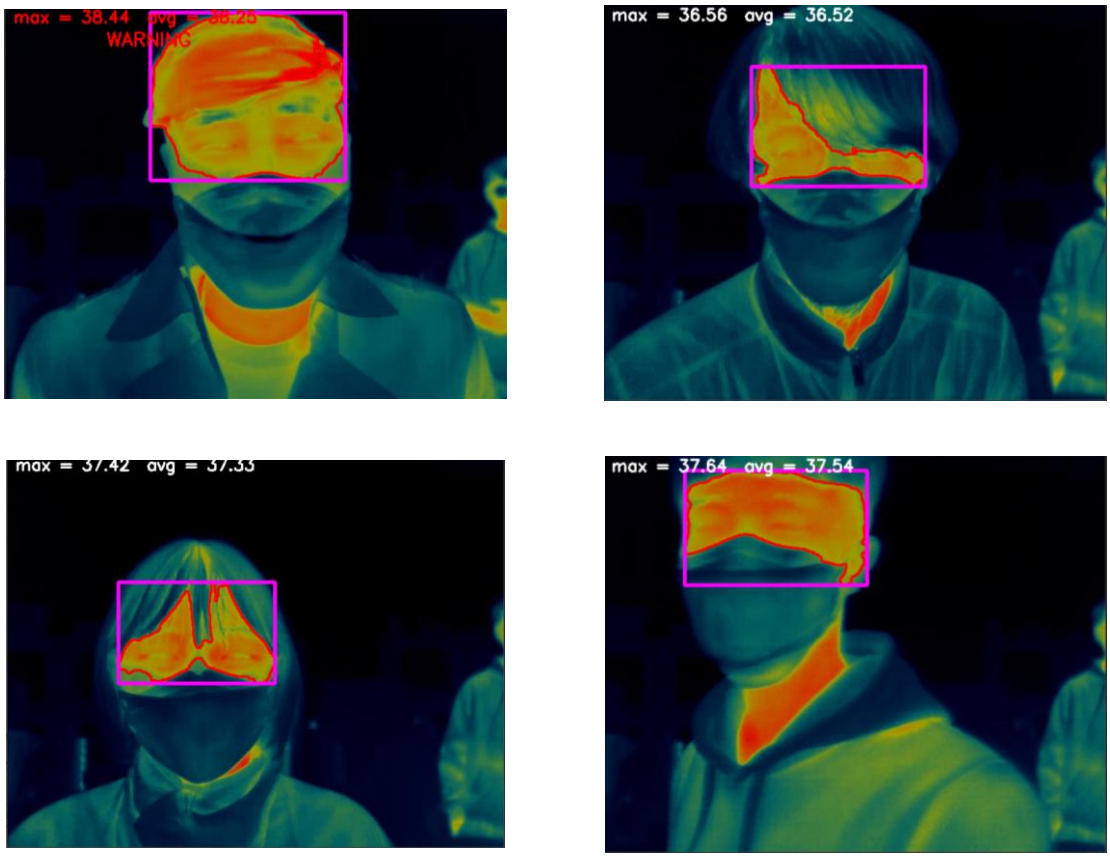


Figure 7. Result image after calculating temperature value.

3. Conclusion

Through this experiment, we studied the process of bringing up values and converting the range of interest (ROI) part of the image to desired values. First, we learned how to transform images by color model and split the channels. In addition, each separated channel values can be mapped to the target values. To obtain the precise target value, we tried to accurately contour a person's face through filtering and morphology. Furthermore, we considered how to establish a mapping function and sorting the values of the ROI. We also found that functions and codes for image processing can vary significantly depending on images, and that the resulting values may not be satisfying, even though the techniques we learned during the session are commonly used for image processing.

4. Appendix (Cord)

```cpp
#include "myOpenCV.h"

using namespace std;
using namespace cv;

int hmin = 0, hmax = 50, smin = 180, smax = 255, vmin = 80, vmax = 215;

int main()
{
    Mat image, image_disp, hsv, hue, mask, dst;
    vector<vector<Point> > contours;
    vector<Vec4i> hierarchy;


    VideoCapture cap("IR_DEMO_cut.avi");

    bool bSuccess = cap.read(image);

    if (!bSuccess)
    {
        cout << "Cannot find a frame from video stream/n";
    }
```

```cpp
    for (;;)
    {
        cap.read(image);
        image.copyTo(image_disp);

        if (image.empty()) {
            break;
        }

        /******** RGB instead of HSV ********/
        cvtColor(image, hsv, COLOR_BGR2HSV);

        vector<Mat> channels;
        split(hsv, channels);

        Mat value(image.size(), CV_8UC1);
        value = channels[2];

        /******** Add Pre-Processing filtering  ********/
        imFilter(&hsv, &hsv, 5, GAUSSIAN);

        /******** set dst as the output of InRange ********/
        inRange(hsv, Scalar(MIN(hmin, hmax), MIN(smin, smax), MIN(vmin, vmax)), Scalar(MAX(hmin, hmax), MAX(smin, smax), MAX(vmin, vmax)), dst);

        /******** Add Post-Processing morphology  ********/
        imMorphology(&dst, &dst, MORPH_RECT, 5, OPENING);

        /********  Find All Contour   ********/
        findContours(dst, contours, hierarchy, CV_RETR_EXTERNAL, CV_CHAIN_APPROX_SIMPLE);
```

```cpp
        if (contours.size() > 0) {

            /// Find the Contour with the largest area ///
            int idx = 0, largestComp = 0;
            double maxArea = 0;
            for (; idx >= 0; idx = hierarchy[idx][0])
            {
                const vector<Point>& c = contours[idx];
                double area = fabs(contourArea(Mat(c)));
                if (area > maxArea)
                {
                    maxArea = area;
                    largestComp = idx; //제일 큰 영역의 컨투어 인덱스
                }
            }

            Rect boxPoint = boundingRect(contours[largestComp]);

            if (boxPoint.x > 70 && boxPoint.x < 270) {
                if (maxArea > 3000) {

                    mask = Mat::zeros(image.size(), CV_8UC1);
                    drawContours(mask, contours, largestComp, Scalar(255), CV_FILLED);
                    bitwise_and(value, mask, value);

                    int size_1D = boxPoint.width * boxPoint.height;
                    Mat array_1D(1, size_1D, CV_8UC1);
```

```
                    for (int y = 0; y < boxPoint.height; y++) {
                        for (int x = 0; x < boxPoint.width; x++) {
                            array_1D.at<uchar>(0, y * boxPoint.width + x) = value.at<uchar>(y + boxPoint.y, x + boxPoint.x);
                        }
                    }

                    cv::sort(array_1D, array_1D, SORT_DESCENDING);

                    double avg_max = 0;
                    double avg = 0;


                    for (int y = size_1D / 100; y < 2 * size_1D / 100; y++) {
                        if (y < size_1D / 100 + 5) avg_max += array_1D.at<uchar>(0, y);
                        avg += array_1D.at<uchar>(0, y);
                    }

                    avg_max = avg_max / 5;
                    avg = avg / (size_1D / 100);

                    avg_max = 4.943047496238989e-05 * pow(avg_max,2) + 0.052213625617881 * avg_max + 25;
                    avg = 4.943047496238989e-05 * pow(avg, 2) + 0.052213625617881 * avg + 25;

                    /// Draw the max Contour on Black background  Image ////
                    drawContours(image_disp, contours, largestComp, Scalar(0, 0, 255), 2, 8, hierarchy); //BGR순서대로 색상

                    /// Draw the Contour Box on Original Image ///
                    rectangle(image_disp, boxPoint, Scalar(255, 0, 255), 3);


                    char savg[20];
                    char savg_max[20];
                    char Massage_max[50] = "max = ";
                    char Massage_avg[50] = "  avg = ";
                    sprintf(savg, "%.2lf", avg);
                    strcat(Massage_avg, savg);
                    sprintf(savg_max, "%.2lf", avg_max);
                    strcat(Massage_max, savg_max);
                    strcat(Massage_max, Massage_avg);
                    if (avg_max > 38) {
                        putText(image_disp, Massage_max, Point(10, 20), FONT_HERSHEY_SIMPLEX, 0.7, Scalar(0, 0, 255), 2);
                        putText(image_disp, "WARNING", Point(130, 50), FONT_HERSHEY_SIMPLEX, 0.8, Scalar(0, 0, 255), 2);
                    }
                    else {
                        putText(image_disp, Massage_max, Point(10, 20), FONT_HERSHEY_SIMPLEX, 0.7, Scalar(255, 255, 255), 2);
                    }

                }
            }
        }

        /// 영상 저장
        //outputVideo << image_disp;
        imshow("tmeprature Measurement", image_disp);

        char c = (char)waitKey(10);
        if (c == 27)
            break;

    }

    return 0;
}
```

5. Peer Evaluation

21500264 Seongryeong Park – Part 1 cording and report

21600372 Yoonkyoung Song – Part 2 cording and report