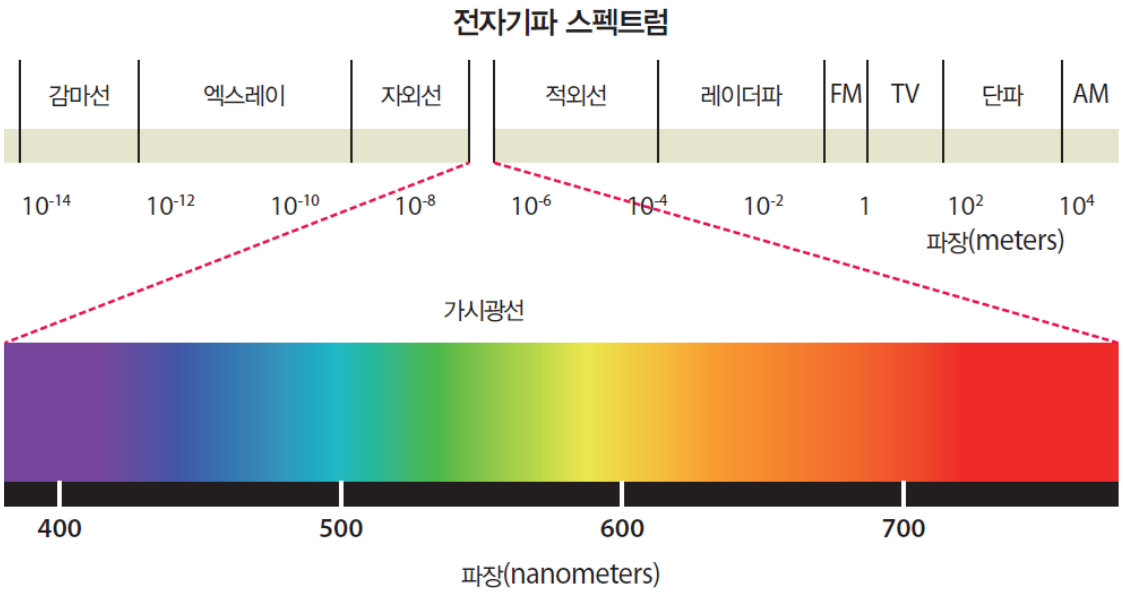
9장 컬러 영상 처리

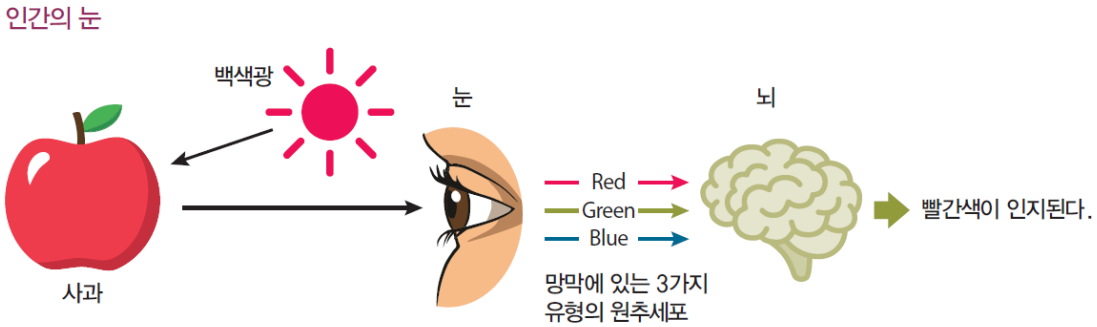
(Color Image Processing)

컬러 개요

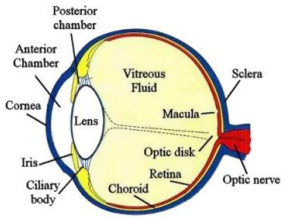
- Spectral theory of light (1672, Issac Newton)
  - Color is a single frequency or wavelength of electro-magnetic radiations
  - Wavelength of visible light: 400 ~ 700 nm



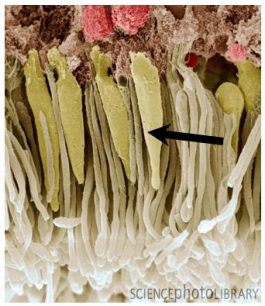
컬러 개요



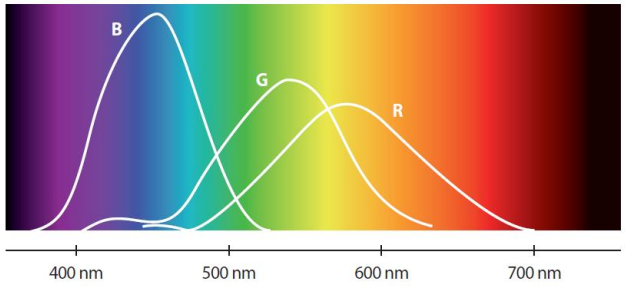
- 삼중 자극 (Tristimulus)
  - 인간의 망막(retina) 내부: 색을 감지하는 3개의 추상체(cone) 존재
    추상체: 빛을 감지하는 3개의 센서로 각각 적(red),녹(green),청(blue)영역 감지
  - 인간의 색 인지 능력은 세가지 추상체들의 반응에 의해 나타남



Human eye



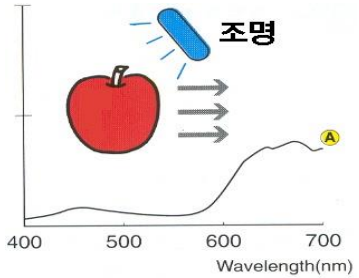
Cone



Tristimulus

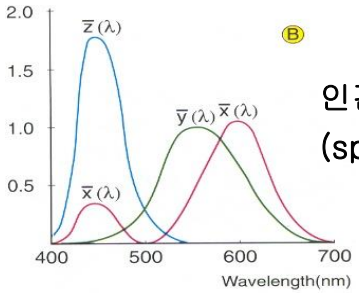
컬러 개요

- 인간의 색 인식 과정
  - 물체의 색 = 물체 고유 특성 (표면특성, 반사도 등) + 조명의 색
  - Tristimulus value = 물체의 스펙트럼 X 표준 광감도
  - 표준 광감도: 국제조명위원회(CIE)가 1931년 정한 표준관찰자(standard observer)의 빛의 파장대별 표준응답특성 그래프



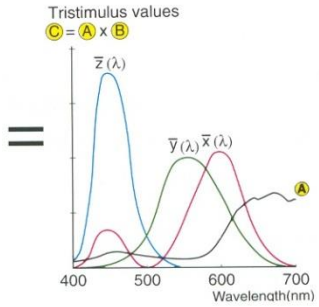
사과의 스펙트럼

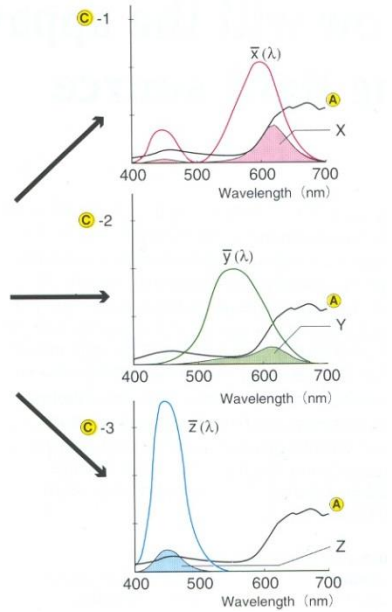
X



인간의 표준 광감도
(spectral sensitivity)

=

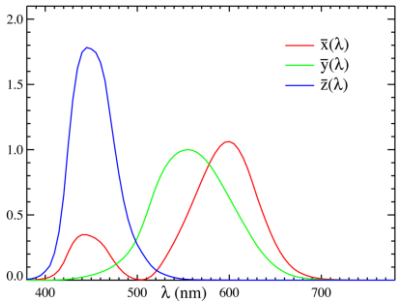




=> 붉은색으로 인식

컬러 개요

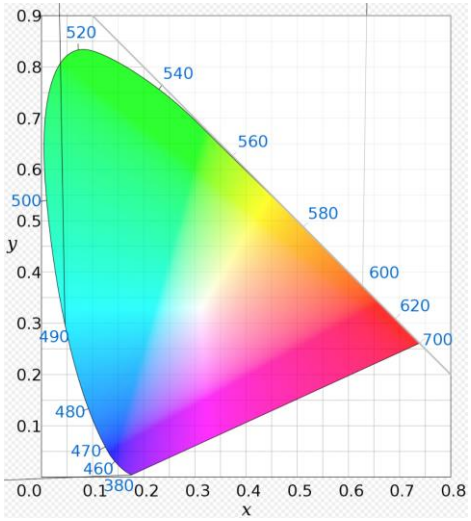
- CIE 1931 색 공간
  - 국제조명위원회(CIE) 가 1931 년 제정한 색공간
  - 인간의 색채 인지에 대한 연구를 바탕으로 수학적으로 정의
  - CIE XYZ 색 공간



$$X = \int_0^\infty I(\lambda)\,\bar{x}(\lambda)\,d\lambda \qquad x = \frac{X}{X+Y+Z}$$

$$Y = \int_0^\infty I(\lambda)\,\bar{y}(\lambda)\,d\lambda \qquad y = \frac{Y}{X+Y+Z}$$

$$Z = \int_0^\infty I(\lambda)\,\bar{z}(\lambda)\,d\lambda \qquad z = \frac{Z}{X+Y+Z} = 1-x-y$$



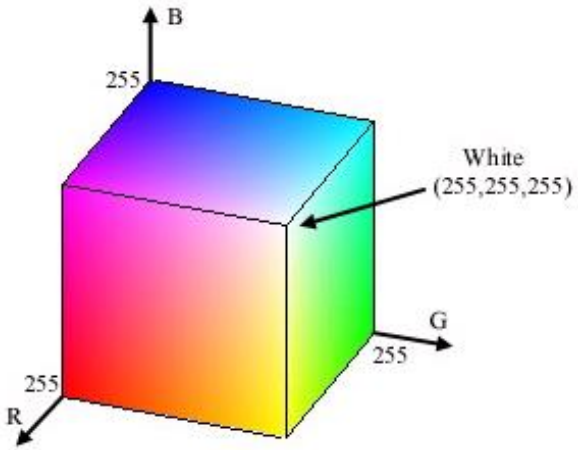
$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \frac{1}{b_{21}} \begin{bmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} = \frac{1}{0.17697} \begin{bmatrix} 0.49 & 0.31 & 0.20 \\ 0.17697 & 0.81240 & 0.01063 \\ 0.00 & 0.01 & 0.99 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

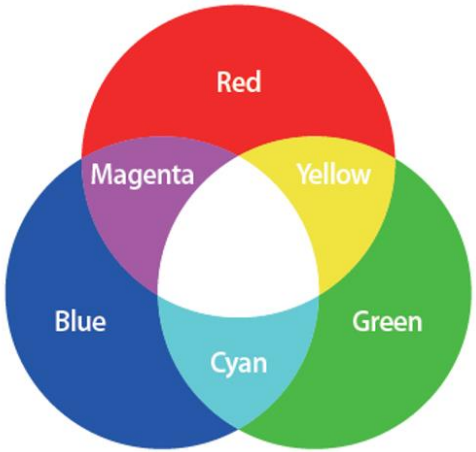
컬러 모델

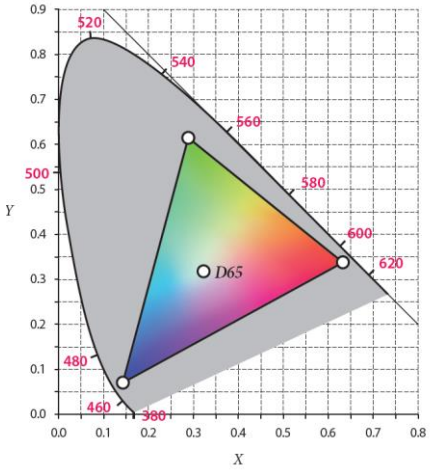
- **RGB 모델**
  - R(red) G (green) B(blue) : 빛의 3원색
  - 각각을 더해 원하는 컬러를 만들어 냄 : additive primaries

    (ex) COLORREF cr = RGB(255,255,0) // 8 bits * 3 = 24 bits

    256 x 256 x 256 = 16,772,216 개
  - CRT 모니터, 컴퓨터 그래픽스 등에서 많이 활용







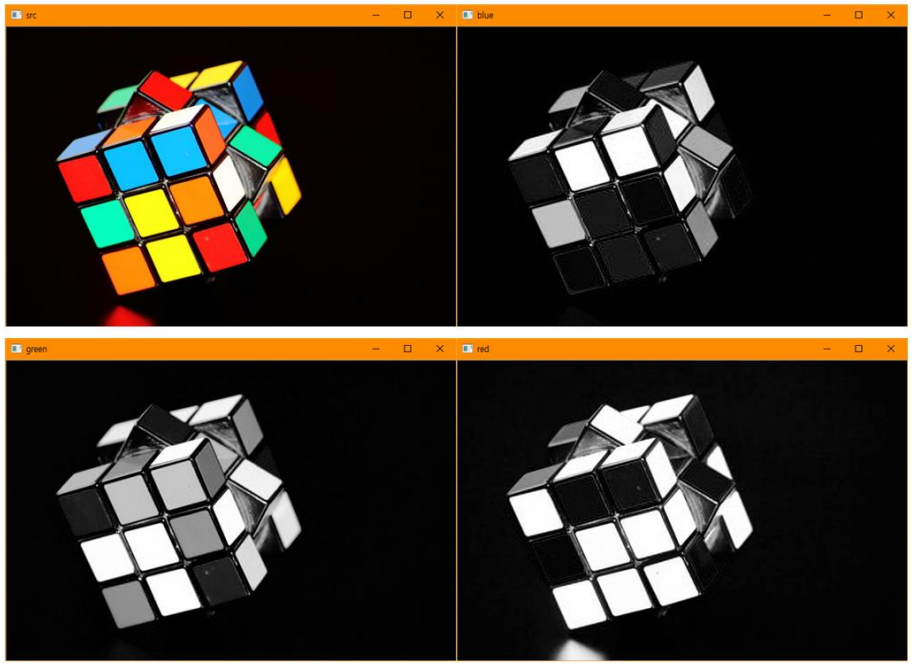
CIE 색 공간 중
RGB 모델로 표현 가능한 영역

실습 1

```
int main()
{
    Mat image;
    image = imread("d:/rcube.jpg", CV_LOAD_IMAGE_COLOR);

    Mat bgr[3];
    split(image, bgr);

    imshow("src", image);
    imshow("blue",bgr[0]);
    imshow("green",bgr[1]);
    imshow("red",bgr[2]);
    waitKey(0);
    return 0;
}
```



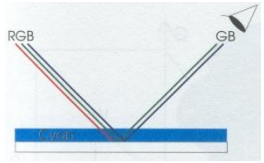
컬러 모델

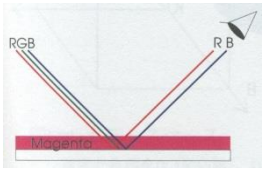
- **CMY 모델**
  - C(Cyan), M(Magnata), Y(Yellow) : 색의 3원색
    (cf) CMYK: K = black
  - RGB 와의 관계
    - C, M , Y color 가 칠해진 표면에 백색광을 비추면 각 각 R, G, B 흡수

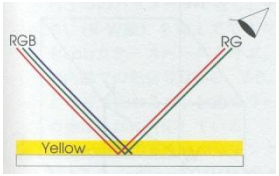
      C = 1 - R (RGB에서 R 성분 제거)

      M = 1 - G (RGB에서 M 성분 제거)

      Y = 1 - B (RGB에서 B 성분 제거)







  - 컬러복사기, 프린터에서 활용

컬러 모델

- **YIQ 모델 (or YUV)**
  - Y: luminance (명암도), I: hue (색상), Q: saturation (채도)

$$\begin{pmatrix} Y \\ I \\ Q \end{pmatrix} = \begin{pmatrix} 0.229 & 0.587 & 0.114 \\ 0.596 & -0.275 & -0.321 \\ 0.212 & -0.528 & 0.311 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}$$

$$\begin{pmatrix} R \\ G \\ B \end{pmatrix} = \begin{pmatrix} 1.000 & 0.956 & 0.621 \\ 1.000 & -0.273 & -0.647 \\ 1.000 & -1.104 & 1.701 \end{pmatrix} \begin{pmatrix} Y \\ I \\ Q \end{pmatrix}$$

  - TV 방송, 영상압축 에서 사용
    - 수신기의 흑백/칼라 관계없이 TV 신호 송출
      - 흑백 TV → Y 신호만 수신
      - 칼라 TV → Y, I, Q 신호 모두 수신 후 R, G, B 변환
  - 장점
    1) 명암도 값을 바로 취할 수 있음: 별도의 영상처리 불필요
    2) 사람의 눈은 컬러값(색상, 채도) 보다 밝기값(명암도)에 더 민감
       - 영상신호 전송 시 Y 신호 덜 압축, I, Q 신호는 더 압축

컬러 모델

YIQ 컬러모델

(원 사진)

(Y)

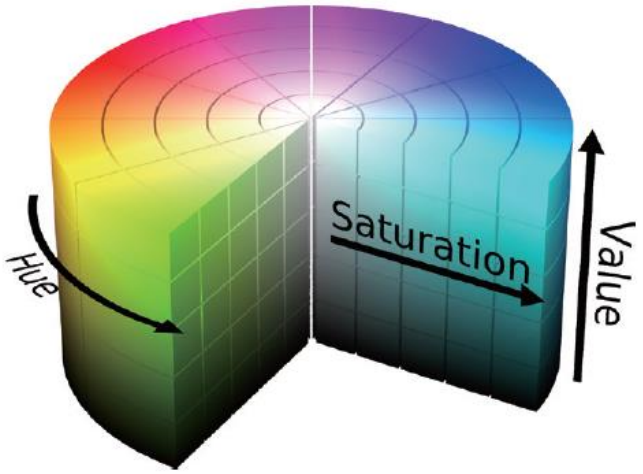
(I)

(Q)

컬러 모델

- **HSV 모델 (HSI)**
  - 인간의 색 인지 방식에 기반을 둔 색상 모형
    (cf) RGB, YIQ, CMY : 하드웨어 기반

  - H (hue) 색상: 빨, 노 ,파 등 색상 표현, (0-360도)
  - S (saturation) 채도: 순색에 첨가된 백색광 비율 (0-1)
  - V (value or intensity) 명도: 빛의 세기 (0-255)



컬러 모델

- RGB to HSV (HSI)

H :색상(Hue)

$$H=\begin{cases}\theta & \text{if } B\le G\\ 360-\theta & \text{if } B\ge G\end{cases}$$

$$\theta=\cos^{-1}\left\{\frac{\frac{1}{2}[(R-G)+(R-B)]}{[(R-G)^2+(R-B)(G-B)]^{1/2}}\right\}$$

S : 채도(Saturation–색상의 밝고 탁함)

$$S=1-\frac{3}{(R+G+B)}[\min(R,G,B)]$$

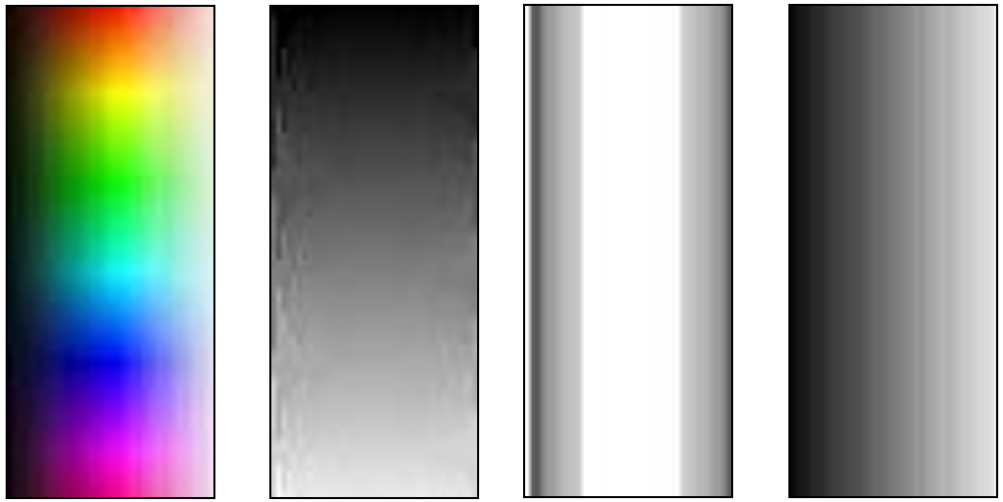
I : 명도(Intensity–밝기의 정도, Value라고도 표기)

$$I=\frac{1}{3}(R+G+B)$$

$$V=\max(R,G,B)$$

컬러 모델

- **HSV 모델**
  - 컬러표현을 위해 색 조합이 불필요: H값 자체가 색상값
  - 컬러영상인식에 많이 활용
    - H 값 → 조명의 영향을 받지 않고 특정 색상 영역을 분리할 때 사용



원본 H S V

컬러 모델

- **HSV 모델**







원본 hue saturation value

컬러 모델

- OpenCV 함수

```
void cv::cvtColor ( InputArray  src,
                    OutputArray dst,
                    int         code,
                    int         dstCn = 0
                  )
```

〈표 6.4.2〉 컬러 공간 변환을 위한 옵션 상수

옵션 상수	값	옵션 상수	값	옵션 상수	값
CV_BGR2BGRA	0	CV_XYZ2RGB	35	CV_BGR2HLS	52
CV_BGRA2BGR	1	CV_BGR2YCrCb	36	CV_RGB2HLS	53
CV_BGR2RGBA	2	CV_RGB2YCrCb	37	CV_HSV2BGR	54
CV_RGBA2BGR	3	CV_YCrCb2BGR	38	CV_HSV2RGB	55
CV_BGR2RGB ,	4	CV_YCrCb2RGB	39	CV_Lab2BGR	56
CV_BGRA2RGBA	5	CV_BGR2HSV	40	CV_Lab2RGB	57
CV_BGR2GRAY	6	CV_RGB2HSV	41	CV_Luv2BGR	58
CV_RGB2GRAY	7	CV_BGR2Lab	44	CV_Luv2RGB	59
CV_GRAY2BGR	8	CV_RGB2Lab	45	CV_HLS2BGR	60
CV_GRAY2BGRA	9	CV_BayerBG2BGR	46	CV_HLS2RGB	61
CV_BGRA2GRAY	10	CV_BayerGB2BGR	47	CV_BGR2YUV	82
CV_RGBA2GRAY	11	CV_BayerRG2BGR	48	CV_RGB2YUV	83
CV_BGR2XYZ	32	CV_BayerGR2BGR	49	CV_YUV2BGR	84
CV_RGB2XYZ	33	CV_BGR2Luv	50	CV_YUV2RGB	85
CV_XYZ2BGR	34	CV_RGB2Luv	51		

실습 2

예제 6.4.3 컬러 공간 변환(BGR→HSV) conver_HSV.cpp

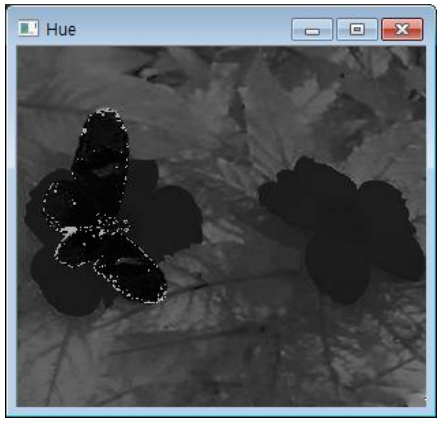
```
01  #include <opencv2/opencv.hpp>
02  using namespace cv;
03  using namespace std;
04
05  void bgr2hsi(Mat img, Mat &hsv)                    // BGR 컬러 → HSI 컬러
06  {
07      hsi = Mat(img.size(), CV_32FC3);               // HSI 행렬 – 3채널, float형
08      for (int i = 0; i < img.rows; i++) {
09          for (int j = 0; j < img.cols; j++)
10          {
11              float B = img.at<Vec3b>(i, j)[0];          // 파란색 화소값
12              float G = img.at<Vec3b>(i, j)[1];          // 녹색 화소값
13              float R = img.at<Vec3b>(i, j)[2];          // 빨간색 화소값
14
15              float s = 1 - 3 * min(R, min(G, B)) / (R + B + G);      // 채도 계산
16              float v = (R + G + B) / 3.0f;                           // 명도 계산
17
18              float tmp1 = ((R - G) + (R - B)) * 0.5f;
19              float tmp2 = sqrt((R - G) * (R - B) + (G - B) * (G - B));
20              float angle = acos(tmp1 / tmp2) * (180.f / CV_PI);
21              float h = (B <= G) ? angle : 360 – angle;               // 색상 계산
22
```

실습 2

```
23                  hsi.at<Vec3f>(i, j) = Vec3f(h / 2, s * 255, v);    // 반환행렬 원소에 지정
24              }
25          }
26          hsi.convertTo(hsi, CV_8U);
27      }
28
29      int main()
30      {
31          Mat BGR_img = imread("../image/color_space.jpg", IMREAD_COLOR);
32          CV_Assert(BGR_img.data);
33          Mat HSI_img, HSV_img, hsi[3], hsv[3];
34
35          bgr2hsi(BGR_img, HSI_img);                          // BGR에서 HSV 변환
36          cvtColor(BGR_img, HSV_img, CV_BGR2HSV);             // OpenCV 함수
37          split(HSI_img, hsi);                                // 채널 분리
38          split(HSV_img, hsv);
39
40          imshow("BGR_img", BGR_img);
41          imshow("Hue", hsi[0]);                          // 사용자 정의함수 이용
42          imshow("Saturation", hsi[1]);
43          imshow("Intensity", hsi[2]);
44          imshow("OpenCV_Hue", hsv[0]);                   // OpenCV 제공함수 이용
45          imshow("OpenCV_Saturation", hsv[1]);
46          imshow("OpenCV_Value", hsv[2]);
47          waitKey();
48          return 0;
49      }
```

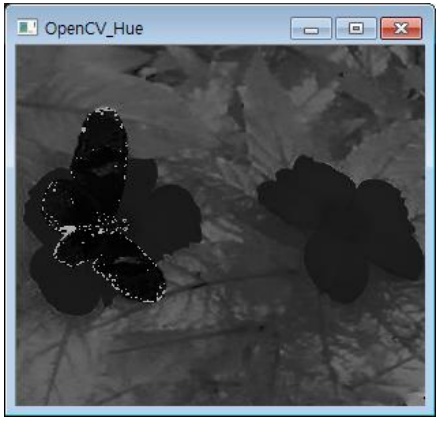
실습 2


BGR_img


Hue


Saturation


Intensity


OpenCV_Hue


OpenCV_Saturation

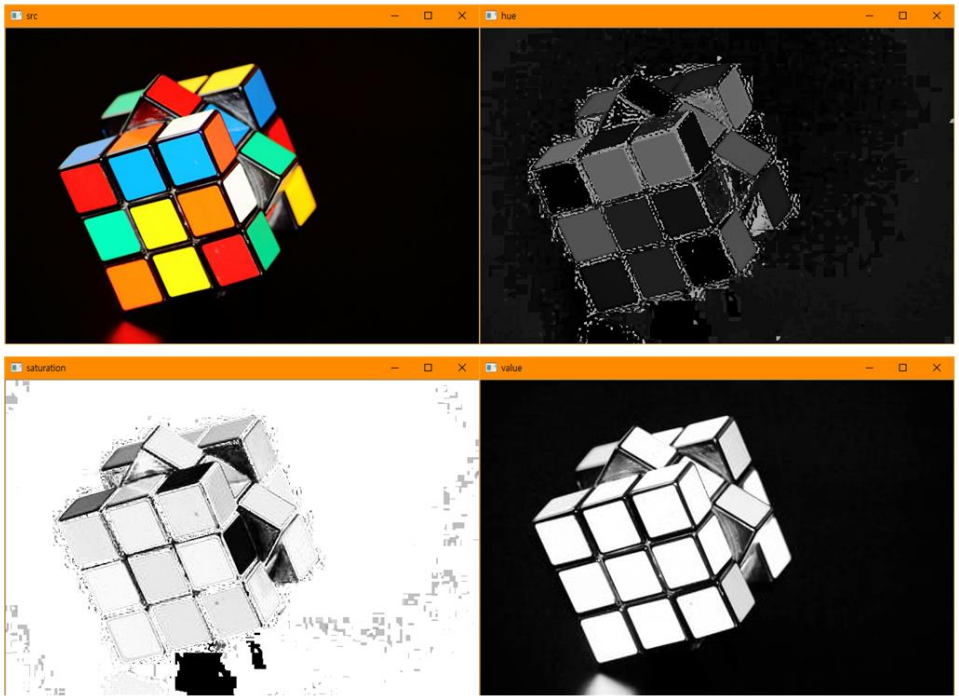

OpenCV_Value

실습 3

```
int main()
{
    Mat image, hsv, dst;
    image = imread("d:/lenna.jpg", CV_LOAD_IMAGE_COLOR);
    cvtColor(image, hsv, CV_BGR2HSV);

    Mat array[3];
    split(hsv, array);
    imshow("src", image);
    imshow("hue", array[0]);
    imshow("saturation", array[1]);
    imshow("value", array[2]);

    waitKey(0);
    return 0;
}
```



실습 4

- 컬러를 이용한 객체 분할

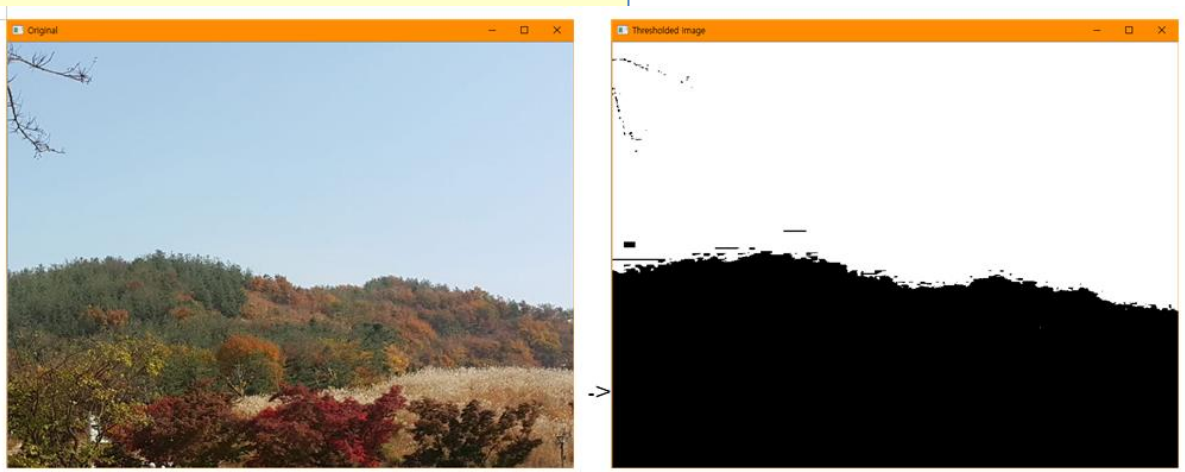
```
int main()
{
    Mat img = imread("d:/image1.jpg", IMREAD_COLOR);
    if (img.empty()) { return -1; }

    Mat imgHSV;
    cvtColor(img, imgHSV, COLOR_BGR2HSV);

    Mat imgThresholded;
    inRange(imgHSV, Scalar(100, 0, 0), Scalar(120, 255, 255), imgThresholded);

    imshow("Thresholded Image", imgThresholded);
    imshow("Original", img);

    waitKey(0);
    return 0;
}
```



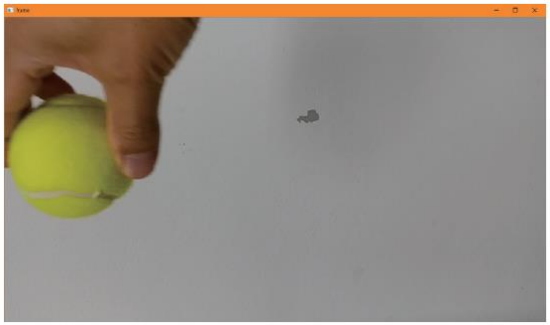
실습 5

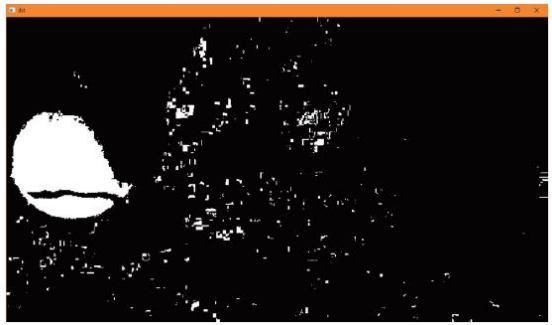
- 컬러를 이용한 객체 추적

```
int main()
{
    VideoCapture cap("d:/tennis_ball.mp4");
    if (!cap.isOpened())
        return -1;
    for (;;)
    {
        Mat imgHSV;
        Mat frame;
        cap >> frame;
        cvtColor(frame, imgHSV, COLOR_BGR2HSV);

        Mat imgThresholded;
        inRange(imgHSV, Scalar(30, 10, 10), Scalar(38, 255, 255), imgThresholded);

        imshow("frame", frame);
        imshow("dst", imgThresholded);
        if (waitKey(30) >= 0) break;
    }
    waitKey(0);
    return 0;
}
```





HW

1. 영상파일을 읽어 윈도우에 표시하고, 마우스 이벤트를 통해서 드래그 할 때 선택된 영역을 새 창에 출력하고, 이 영역에 대해서 Hue (색상) 채널 히스토그램 그래프를 그리는 프로그램을 작성하라.

2. 카메라로 입력되는 동영상에 대하여, 색상을 이용하여 얼굴을 검출(추적) 하는 프로그램을 작성하라