Nota: Algunas de las imágenes que aparecen en esta presentación provienen del libro:
Visión por Computador: fundamentos y métodos.
Arturo de la Escalera Hueso. Prentice Hall.

Intelligent Systems Lab

Sistemas de Percepción

Visión por Computador

Arturo de la Escalera
José María Armingol
Fernando García
David Martín
Abdulla Al-Kaff

cc BY NC SA

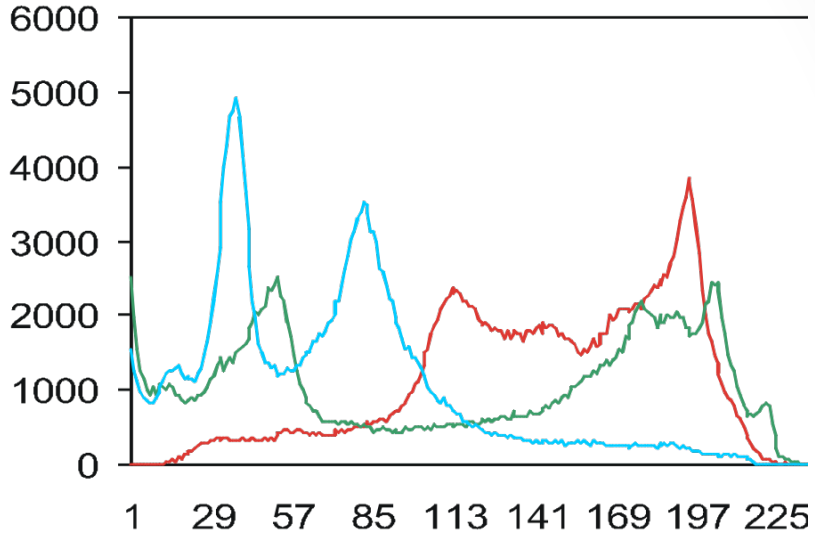
uc3m | Universidad Carlos III de Madrid

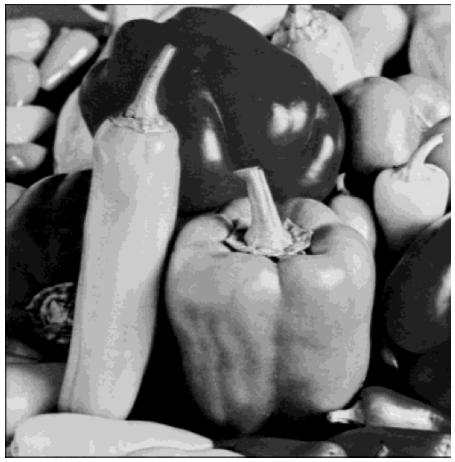
Intelligent Systems Lab

OpenCV : Espacio de color RGB y espacio de color HSV

8/29/2017

Espacio de color RGB





Espacio de color RGB

- Ejemplo 01: Mostrar por pantalla el número de canales:

```
10      // Object instantiation
11      Mat gray_img, color_img;
12      // Load image from disk
13      gray_img = imread("mandril.jpg", CV_LOAD_IMAGE_GRAYSCALE);
14      if (!gray_img.data){
15          cout << "error loading image" << endl;
16          return 1;
17      }
18      color_img = imread("mandril_c.jpg", CV_LOAD_IMAGE_COLOR);
19      if (!color_img.data){
20          cout << "error loading image" << endl;
21          return 1;
22      }
23      cout << "gray channels: " << gray_img.channels() << endl;
24      cout << "color channels: " << color_img.channels() << endl;
25      // Create window canvas to show image
26      namedWindow("gray", CV_WINDOW_AUTOSIZE);
27      namedWindow("color", CV_WINDOW_AUTOSIZE);
28      // Show image in the name of the window
29      imshow("gray", gray_img);
30      imshow("color", color_img);
31      // Function for show the image in ms.
32      waitKey(0);
33      // Free memory
34      destroyAllWindows();
35      // End of the program
36      return 0;
37  }
```

Espacio de color RGB

- Ejemplo 02: Separar la imagen de color y guardar cada uno de los planos de color como imágenes en escala de grises:

split

Divides a multi-channel array into several single-channel arrays.

C++: void **split**(const Mat& **src**, Mat* **mvbegin**)

C++: void **split**(InputArray **m**, OutputArrayOfArrays **mv**)

Parameters:
- **src** – input multi-channel array.
- **mv** – output array or vector of arrays; in the first variant of the function the number of arrays must match `src.channels()`; the arrays themselves are reallocated, if needed.

The functions `split` split a multi-channel array into separate single-channel arrays:

cvtColor

Converts an image from one color space to another.

C++: void **cvtColor**(InputArray **src**, OutputArray **dst**, int **code**, int **dstCn**=0)

Parameters:
- **src** – input image: 8-bit unsigned, 16-bit unsigned (`CV_16UC...`), or single-precision floating-point.
- **dst** – output image of the same size and depth as `src`.
- **code** – color space conversion code (see the description below).
- **dstCn** – number of channels in the destination image; if the parameter is 0, the number of the channels is derived automatically from `src` and `code` .

The function converts an input image from one color space to another. In case of a transformation to-from RGB color space, the order of the channels should be specified explicitly (RGB or BGR). Note that the default color format in OpenCV is often referred to as RGB but it is actually BGR (the bytes are reversed). So the first byte in a standard (24-bit) color image will be an 8-bit Blue component, the second byte will be Green, and the third byte will be Red. The fourth, fifth, and sixth bytes would then be the second pixel (Blue, then Green, then Red), and so on.

Espacio de color RGB

- Ejemplo 02: Separar la imagen de color y guardar cada uno de los planos de color como imágenes en escala de grises:

```
10      // Object instantiation
11      Mat color_img;
12      // Load image from disk
13      color_img = imread("mandril_c.jpg", CV_LOAD_IMAGE_COLOR);
14      if (!color_img.data){
15          cout << "error loading image" << endl;
16          return 1;
17      }
18      // change from bgr to rgb
19      cvtColor(color_img, color_img, CV_BGR2RGB);
20      // Split color image. Required containers
21      Mat r_channel(color_img.size(), CV_8UC1);
22      Mat g_channel(color_img.size(), CV_8UC1);
23      Mat b_channel(color_img.size(), CV_8UC1);
24      Mat array_channels[] = { r_channel, g_channel, b_channel };
25      split(color_img, array_channels);
```

Espacio de color RGB

- Ejemplo 02: Separar la imagen de color y guardar cada uno de los planos de color como imágenes en escala de grises:

```
25      split(color_img, array_channels);
26      // Create window canvas to show image
27      namedWindow("color", CV_WINDOW_AUTOSIZE);
28      namedWindow("R", CV_WINDOW_AUTOSIZE);
29      namedWindow("G", CV_WINDOW_AUTOSIZE);
30      namedWindow("B", CV_WINDOW_AUTOSIZE);
31      // Show image in the name of the window
32      imshow("color", color_img);
33      imshow("R", r_channel);
34      imshow("G", g_channel);
35      imshow("B", b_channel);
36      // Function for show the image in ms.
37      waitKey(0);
38      // Free memory
39      destroyAllWindows();
40      // End of the program
41      return 0;
42  }
```

Espacio de color HSV

- Ejemplo 03: Cargar la imagen a color en formato bgr (por defecto en opencv) y convertirla al espacio de color

```
1   // e01_hsv.cpp: HSV histograms
2   #include "opencv\cv.hpp"
3   #include <iostream>
4
5   using namespace cv;
6   using namespace std;
7
8   int main(int argc, char* argv[])
9   {
10      // Object instantiation
11      Mat color_img, hsv_img;
12      // Load image from disk
13
14      color_img = imread("mandril_c.jpg", CV_LOAD_IMAGE_COLOR);
15      if (!color_img.data){
16          cout << "error loading image" << endl;
17          return 1;
18      }
19      // convert from bgr to hsv
20      cvtColor(color_img, hsv_img, CV_BGR2HSV);
```

Espacio de color HSV

- Separar (Split) cada canal h, s y v

```
21      // Split channels
22      Mat h_channel(hsv_img.size(), CV_8UC1);
23      Mat s_channel(hsv_img.size(), CV_8UC1);
24      Mat v_channel(hsv_img.size(), CV_8UC1);
25      Mat array_channels[] = { h_channel, s_channel, v_channel };
26      split(hsv_img, array_channels);
```

Calcular el histograma para cada canal

```
27      // Initialize histogram parameters
28      /// Establish the number of bins
29      int histSize = 256;
30      /// Set the ranges ( for B,G,R) )
31      float range[] = { 0, 256 };
32      const float* histRange = { range };
33
34      // Calculate histogram
35      Mat h_hist, s_hist, v_hist;
36      calcHist(&h_channel, 1, 0, Mat(), h_hist, 1, &histSize, &histRange, true, false);
37      calcHist(&s_channel, 1, 0, Mat(), s_hist, 1, &histSize, &histRange, true, false);
38      calcHist(&v_channel, 1, 0, Mat(), v_hist, 1, &histSize, &histRange, true, false);
```

Espacio de color HSV

- Preparar la imagen para mostrar los histogramas

```
40      // Plot histogram
41      int hist_w = 512; int hist_h = 400;
42      int bin_w = cvRound((double)hist_w / histSize);
43      Mat histImage(hist_h, hist_w, CV_8UC3, Scalar(0, 0, 0));
44
45      /// Normalize the result to [ 0, histImage.rows ]
46      normalize(h_hist, h_hist, 0, histImage.rows, NORM_MINMAX, -1, Mat());
47      normalize(s_hist, s_hist, 0, histImage.rows, NORM_MINMAX, -1, Mat());
48      normalize(v_hist, v_hist, 0, histImage.rows, NORM_MINMAX, -1, Mat());
49
50      /// Draw for each channel
51      for (int i = 1; i < histSize; i++)
52      {
53          line(histImage, Point(bin_w*(i - 1), hist_h - cvRound(h_hist.at<float>(i - 1))),
54              Point(bin_w*(i), hist_h - cvRound(h_hist.at<float>(i))),
55              Scalar(255, 0, 0), 2, 8, 0);
56          line(histImage, Point(bin_w*(i - 1), hist_h - cvRound(s_hist.at<float>(i - 1))),
57              Point(bin_w*(i), hist_h - cvRound(s_hist.at<float>(i))),
58              Scalar(0, 255, 0), 2, 8, 0);
59          line(histImage, Point(bin_w*(i - 1), hist_h - cvRound(v_hist.at<float>(i - 1))),
60              Point(bin_w*(i), hist_h - cvRound(v_hist.at<float>(i))),
61              Scalar(0, 0, 255), 2, 8, 0);
62      }
```

Espacio de color HSV

- Mostrar todas las imágenes, liberar memoria y finalizar el programa

```
63      // Windows for all the images
64      namedWindow("color", CV_WINDOW_AUTOSIZE);
65      namedWindow("H", CV_WINDOW_AUTOSIZE);
66      namedWindow("S", CV_WINDOW_AUTOSIZE);
67      namedWindow("V", CV_WINDOW_AUTOSIZE);
68      namedWindow("histogram: Red=H Green=S Blue=V", CV_WINDOW_AUTOSIZE);
69
70      // Show image in the name of the window
71      imshow("color", color_img);
72      imshow("H", h_channel);
73      imshow("S", s_channel);
74      imshow("V", v_channel);
75      imshow("histogram: Red=H Green=S Blue=V", histImage);
76
77      // Function for show the image in ms.
78      waitKey(0);
79      // Free memory
80      destroyAllWindows();
81      // End of the program
82      return 0;
83  }
```