Colour Segmentation

Colour segmenting is perhaps one of the simplest computer vision processes. Each pixel in an image has associated with it three values between 0 and 255 (for an 8-bit image, since $2^8 = 256$). Each value indicates the intensity for that pixel's red, green and blue colours, with the final pixel colour determined by a linear combination of these three values.

Let's extract one pixel from MiRo's collar in below image and show its RGB values:

As expected, there is a stronger intensity of the red pixel value vs. green and blue.

When we segment the image according to some RGB boundaries, we're selecting only those pixels with which the RGB values exist within those boundaries.

E.g., let's set a lower boundary of R = 0, G = 0, B = 0 and an upper boundary of R = 255, G = 150, B = 150. Our boundaries are therefore:

Lower limit = [0, 0, 0]
Upper limit = [255, 150, 150]

We pass through each pixel in the image, test whether it's three values exist within this range, and remove (i.e. set its pixel value to [0, 0, 0]) any pixel outside of this range. What should we expect to see with the above limits applied? Try it in the GUI!

Colour segmentation can be a useful pre-processing task in order to extract regions of interest for which further processing can be performed on.

Edge Detection

The edge detector deployed here is called the Canny Edge Detector, after its founder John Canny in 1986. It consists of 3 principle steps (often more can be used but these are the most important). The steps are as follows:

1. *Filter out noise.* To filter out noise in an image, a number or convolutions can be used. In this case the Gaussian filter is employed (see the help page on smoothing and blurring).

2. *Intensity gradients*. This involves determining how the intensity of pixel values change as you move across the pixels in an image. The method used for this operation is called Sobel, and the gradients in the x and y directions are computed as follows:

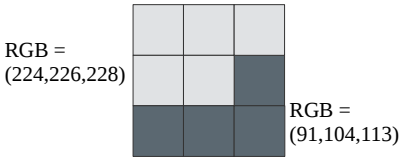
$$G_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix}, G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix}$$

With this, the gradient strength and direction can be found:

$$G = \sqrt{G_x^2 + G_y^2} \quad \theta = \arctan\left(\frac{G_y}{G_x}\right)$$

Direction is rounded to the nearest multiple of 45º (i.e. 0º, 45º, 90º or 135º).

E.g. let us take a 3x3 window at a point that marks the boundary between MiRo's white outer shell and its grey underbelly. We may have the following (simplified and with lines added to differentiate pixels):



in which there are two distinct colours with RGB values as shown above. We have therefore 3 *planes*, or dimensions, with which we could perform the operation on, relating to each colour channel (alternatively one could convert the image first to grayscale and then perform these operations. This would mean on one colour space to compute with). Let's try with the red colour channel. Performing the convolutions as proposed above would give:

$$G_x = \begin{bmatrix} -1 \times 224 & 0 \times 224 & +1 \times 224 \\ -2 \times 224 & 0 \times 224 & +2 \times 91 \\ -1 \times 91 & 0 \times 91 & +1 \times 91 \end{bmatrix} = -266$$

$$G_y = \begin{bmatrix} -1 \times 224 & -2 \times 224 & -1 \times 224 \\ 0 \times 224 & 0 \times 224 & 0 \times 91 \\ +1 \times 91 & +2 \times 91 & +1 \times 91 \end{bmatrix} = -308$$

$$G = \sqrt{(-266)^2 + (-308)^2} = 407$$

$$\theta = \arctan\left(\frac{308}{266}\right) = 45^o \ (rounded)$$

3. *Thresholding*. The two thresholds you entered in the main window are used here:
   a) If pixel gradient is above the upper threshold, it's considered an edge.
   b) If pixel gradient is below the lower threshold, it's rejected.
   c) If pixel gradient is between the two thresholds, it will only be considered an edge if a neighbouring pixel is above the threshold.

There are various ways in which edge detection can be performed, and a number of factors can affect the final result, from the selection of the smoothing operation, convolution weights, threshold boundaries, as well as the colour channels and/or whether you choose to operate on the greyscale image.