Task 1: Detecting the Colored Objects along with number of objects under each color

Javier Galindos Omar El Nahhas

19/09/2021

Abstract

The first task for the Machine Vision course (EEM0040) is to identify colours and count the amount of pills per colour. The presented work is developed in both Python (using OpenCV2) and MATLAB.

1 Methodology

Pre-processing steps have been conducted in order to go from a raw RGB-image to gaining insights into how many pills from which colour are in the image. In order to test the selection and accuracy

Figure 1: Test input image (640x480) with blue, orange and yellow pills.

of the pre-processing methods, a sample input image (Fig.1) was used to test the robustness of the algorithms under non-optimal conditions. The artificial lighting remained the same, but the position of the pills, distance from the camera and the angle of the camera all were changed in the final test of the algorithms.

1.1 Defining colour boundaries

In order to detect colours, it is needed to specify lower and upper threshold for the range in which each colour falls considering current lighting conditions. The boundaries can be found in Table 1 in BGR-format (OpenCV2 standard).

	Lower range	**Upper range**
Blue	[80, 75, 20]	[140, 125, 60]
Orange	[0, 40, 140]	[10, 70, 190]
Yellow	[20, 100, 120]	[80, 165, 180]

Table 1: BGR boundaries for different colors

Using masks with these boundaries, the output is a new image that only contains the data which corresponds to a specific color in the defined range of blue, orange or yellow. This results in three separate images, one for each mask.

1.2 Filtering

- **Gaussian blur**: Kernel size = 15 x 15. $\sigma = 2$. First, Gaussian blur is applied to make the image blurry, essentially 'smearing out' pixels to remove random noise from the image.
- **Erosion**: Kernel size = 3 x 3. Secondly, erosion is applied to remove small islands of noisy pixels, while also reducing the size of bigger components to avoid confusion for counting the connected components.
- **Noise reduction**: Finally, a Non-local Means Denoising algorithm is applied to the image. This filter takes the mean of all pixels in the image and are weighted by the similarity to the current pixel, which compared to Local Means Denoising is able to hold on to more detail in the output.

1.3 Thresholding

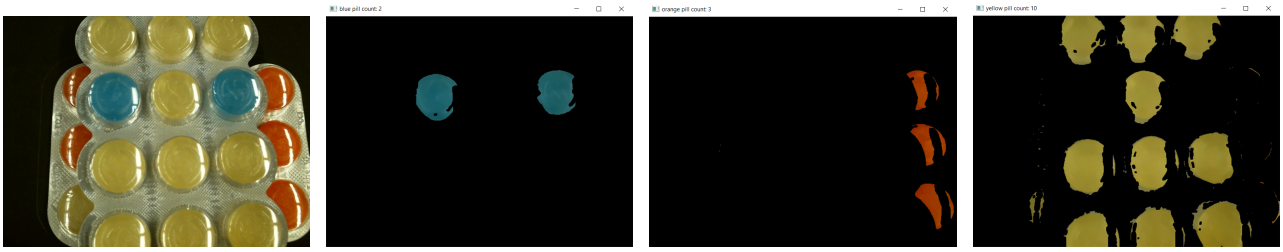
After applying the pre-processing and filtering techniques, binary thresholding is applied on the images in order to conform with input formatting of the function required to find object contours. Binary thresholding sets every pixel < 127 to 0, and $>= 127$ to 1, generating a black and white image.

1.4 Counting connected components

Finally, connected components are detected within the binary images. To avoid taking into account remaining noise in the counting, a threshold of 1500 pixels is applied for the minimum area to be identified as a pill. This threshold was set through trial and error.

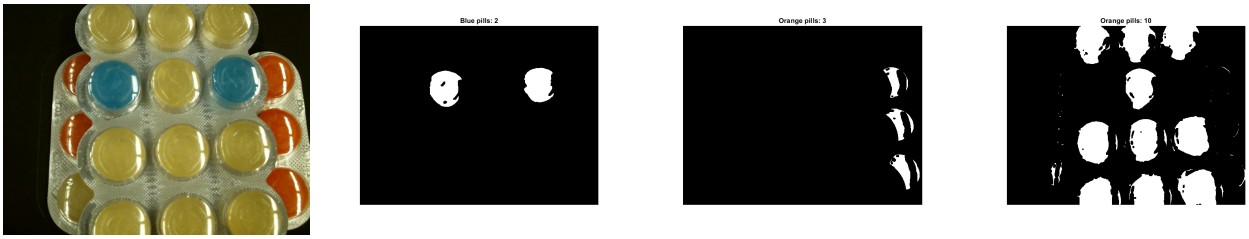
2 Results

The input image and its output in Python and Matlab can be seen in Fig.2 and Fig.3 respectively. Using the same type of filters with the same parameters (within capability), the output is the same in both programming languages. Both programs recognize the amount of pills correctly in the top pack of pills, which have been rotated 90° compared to the orientation the algorithms were tuned on. Pills in the bottom pack get correctly recognized and counted on the right side. However, pills on the left side suffer significantly different lighting features, causing the pills on the left side of the bottom pack to not be recognized.



(a) Original image (b) Blue pills: 2 (c) Orange pills: 3 (d) Yellow pills: 10

Figure 2: Results using Python and OpenCV2.



(a) Original image (b) Blue pills: 2 (c) Orange pills: 3 (d) Yellow pills: 10

Figure 3: Results using MATLAB.

3 Discussion

From this work, it is worth to observe the difference between Python and Matlab results when performing the same pre-processing algorithms with the same parameters (within capability). The OpenCV2 functions perform better at noise removal, resulting in 'fuller' pills and therefore more accurate counting on a larger scale (see Fig.2b vs. Fig.3b). This difference can be explained partly through the arithmetical difference in calculations in both languages, and partly through different implementation of the pre-processing algorithm functions. Due to the scope of this assignment, those differences will not be further researched.