Task 2: Find the locations of objects and counting the Number of Objects

Javier Galindos (214373IV) Omar El Nahhas (214316IV)

02/10/2021

Abstract

The second task for the Machine Vision course (EEM0040) is to find the locations of objects and counting the number of them and measure the distance between pills and to identify the colour of those pills. The presented work is developed in both Python (using OpenCV2) and MATLAB.

1 Methodology

In this assignment, a pill-card with empty and filled slots are presented. The goal is to locate and identify the objects, and then measuring the Manhattan distance from the objects to a defined origin. Pattern matching with four templates (empty, blue, yellow, orange) is applied to identify the relevant pill and its centre.

A raw camera image from a non-optimal position cannot be used to measure real-world distances. In order to convert pixels [px] into millimetres [mm], the camera needs to be calibrated accordingly, aiming to determine its geometric parameters of the image formation process. Using 10 images of a chessboard in various orientations/distances and a native OpenCV2 function, the captured image is transformed with the calculated distortion matrices and shows the true representation of the image, now enabling the conversion between pixels and millimetres. Due to the controlled environment of this application, this form of calibration will not be used from this point onward. Rather, the camera is situated directly above the image. This avoids to correct for perspective, which make the distortion corrections negligible. Therefore, the pixel/mm ratio can directly be derived from the source image in this use-case.

1.1 Template matching

Both in Python and Matlab, Normalized Cross-Correlation (NCC) was used to identify the locations with the highest probabilities of the template matching with the to-be-analysed image. In Python, template matching results in start and end coordinates of a rectangle. Matlab returns single coordinates, where the highest probabilities have a higher pixel intensity, see Fig. 4. Both implementations

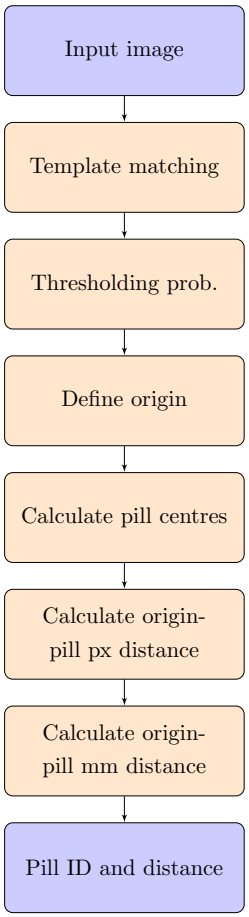

Figure 1: Flowchart of the image processing methodology.

allowed for template matching with all three RGB channels, which makes object detection increasingly more accurate than having to choose one channel to differentiate from. On the professor's suggestion, HSV channels were also considered. Again, one HSV channel could not compete with the level of differentiation that three RGB channels could offer, hence this one-channel approach was rejected.

1.2 Finding the object's centre

- **Python**: Boundaries are created by saving all coordinates with a probability > 0.6, found through trial & error. Then, Non-Maximum Suppression is applied to avoid many overlapping rectangles on the same pill. This application results in only the rectangles with the highest probability making it to the final selection. The remaining rectangles are then used to estimate the centre of the pill, which is the mean of the rectangle in the X and Y plane. Because the camera set-up is in a controlled environment, the true means of the pills are known (see Figure 5). Next, the K-Nearest Neighbour (K-NN) algorithm is applied to couple every calculated centre to its nearest ground-truth mean. Doing this, the exact centre and color of the pill are

Figure 2: Test input image.

(a) Empty template. (b) Orange template. (c) Blue template. (d) Yellow template.

Figure 3: Templates of the different pills.

now known parameters.

- **Matlab**: The probabilities which result from Matlab's template matching function are cut off at > 0.8, which is found through trial & error. This results in many points being identified at a similar location, see Fig. 6. This problem could be solved using Non-Maximum Suppression as well, but since Matlab does not have a native function to perform this quickly and cleanly, it is decided to perform the DBSCAN algorithm to look for all the clusters and pick only one point to be its mean for every cluster. This point is then considered as the centre of the pill to be calculated with. Since the resulting centres are seemingly perfectly centered, there is no K-NN applied to rectify the centres.

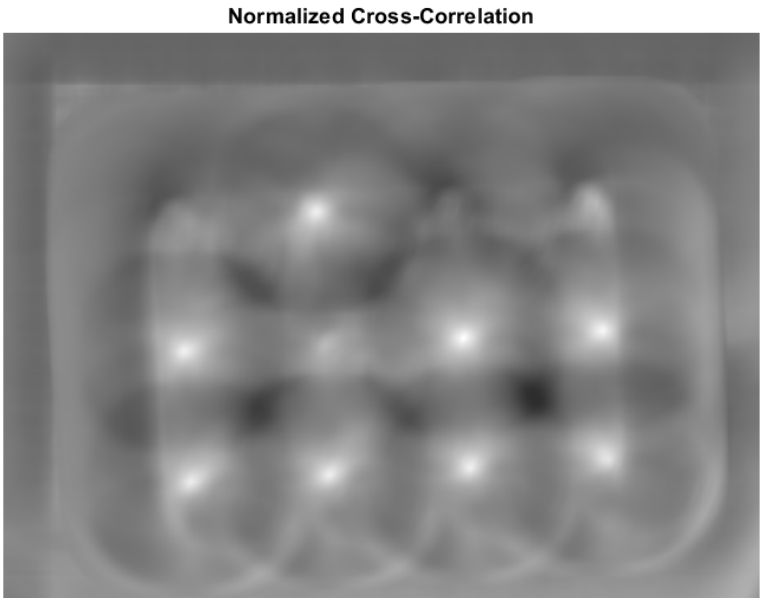

Figure 4: Normalized cross-correlation of the Test input.

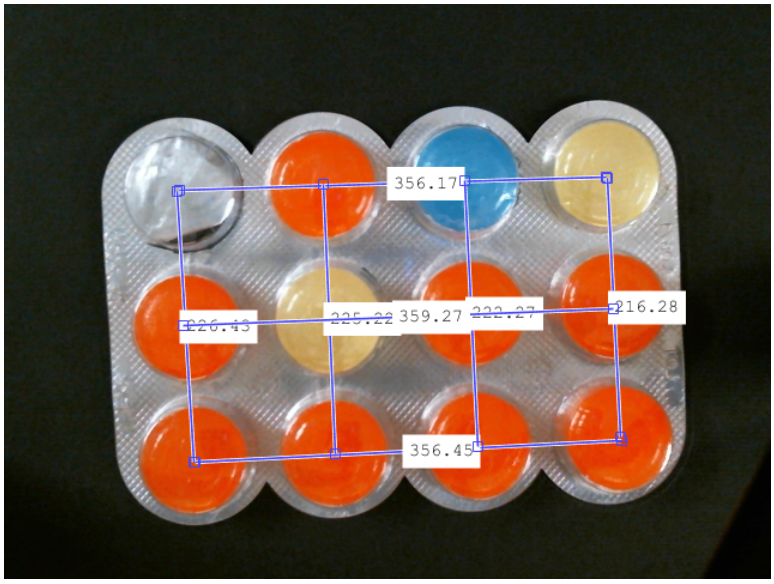

Figure 5: Test input image with a grid to identify the real mean values.

1.3 Calculating Manhattan distance

For computing the Manhattan or City-Block distance from the origin pill to the other pills, the approximate center of the pill is found and the distance between pills is computed as the difference between the centers of the pill. The pixel/mm ratio is found through measuring the amount of pixels of the whole package, divided by the real-life measurement in millimetres. The package covers 106 mm and 473.31 px (see Fig. 7, resulting in a px/mm ratio of 106/473.31.

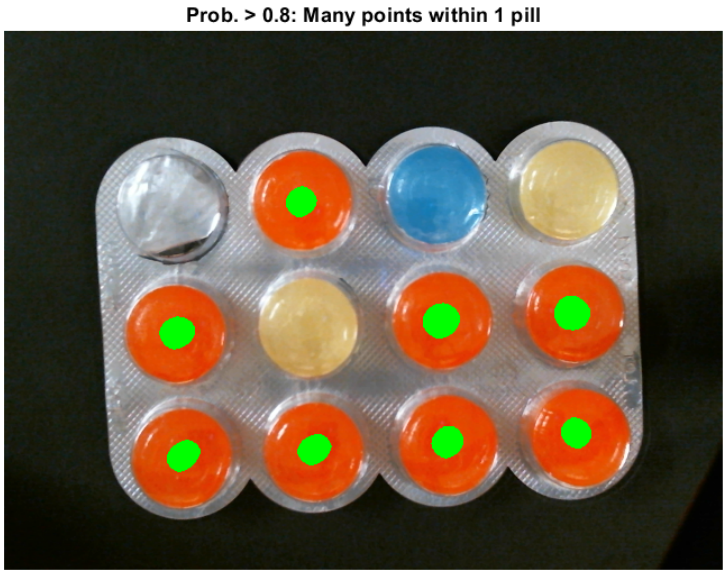

Figure 6: Test input image with NCC probabilities.

Figure 7: Test input image with pixel to mm ratio.

2 Results

In Fig. 8 and Fig. 9 the results of the pill identification and distance measurement can be observed. Python's pill centers have been rectified to its ground-truth using K-NN, while Matlab's pill centers are a direct result of the DBSCAN centroid of that particular cluster of points. The real-life measured distances are approximately 26 mm from the center-to-center distance of two pills, making Python and Matlab's results highly accurate.

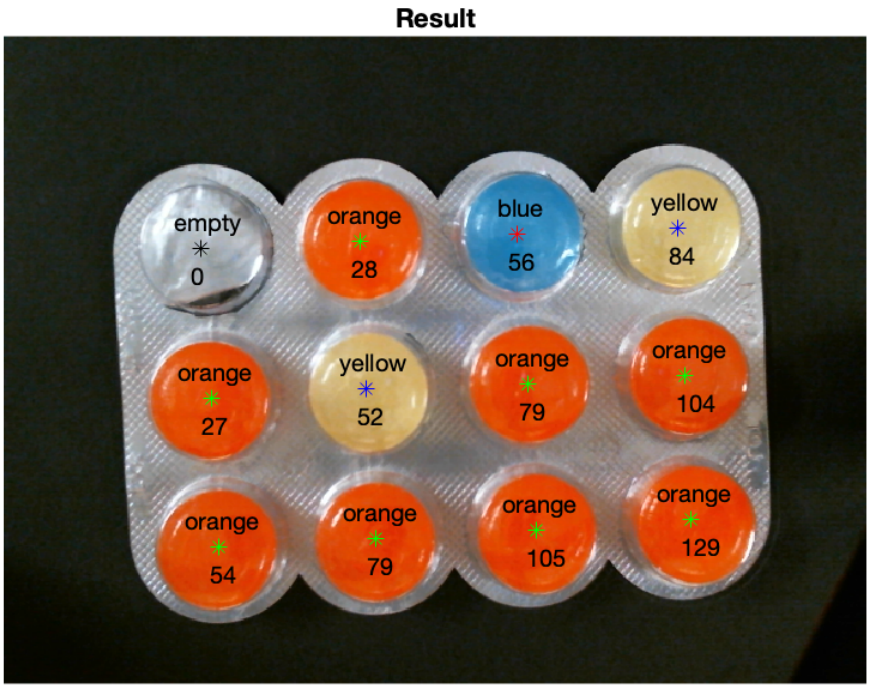

Figure 8: Test input image with colors of the pills identified and distance to the origin measured in mm (MATLAB).

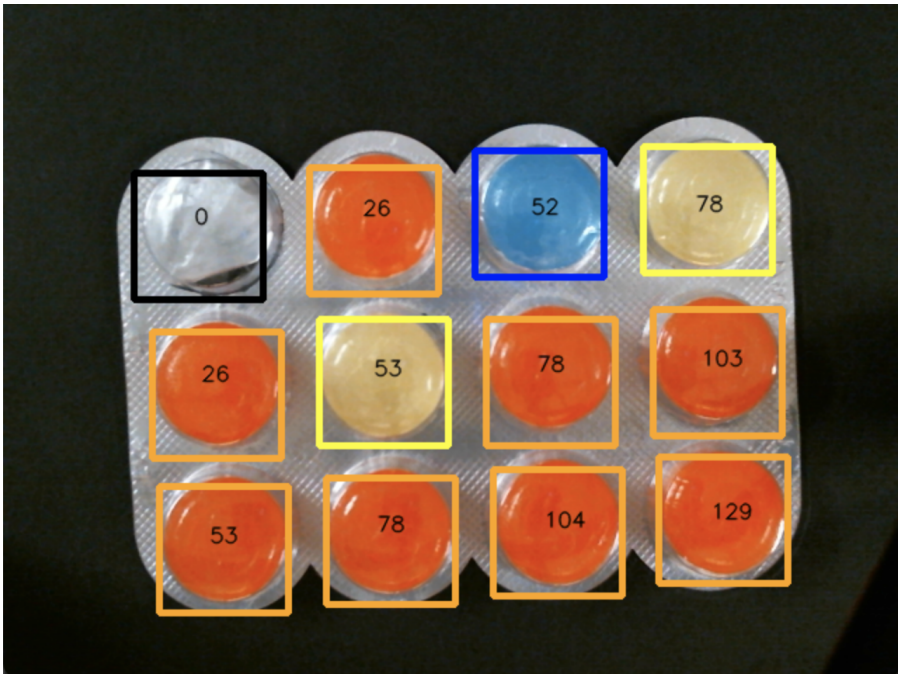

Figure 9: Test input image with colors of the pills identified and distance to the origin measured in mm (Python).

3 Discussion

The limitations of the template matching approach to acquire the colour and center of the pills are related to the camera positioning and the room lighting. Without changing the templates, but with a new input image, the results are seen in Fig. 10. Here, it is seen that the algorithm does not break, it chooses a new origin and computes the distances from there. Pills in the top corner suffer the biggest differences in lighting, hence the lack of recognition at the aforementioned probability cut-offs. This approach has only been tested in artificially maintained environments, and is therefore not expected to be robust in any other condition due to its limitations with the pattern matching algorithms. If conditions were to change, the camera should be calibrated according to the method mentioned in the methodology. To make the pattern matching more robust, deep-learning networks can be used with multiple pill templates to recognize the pills at different orientations, sizes and lighting conditions. If no deep-learning/additional templates can be used, the current algorithm can be improved by applying a Gaussian pyramid to obtain the same template at several scales, which will improve template matching accuracy.

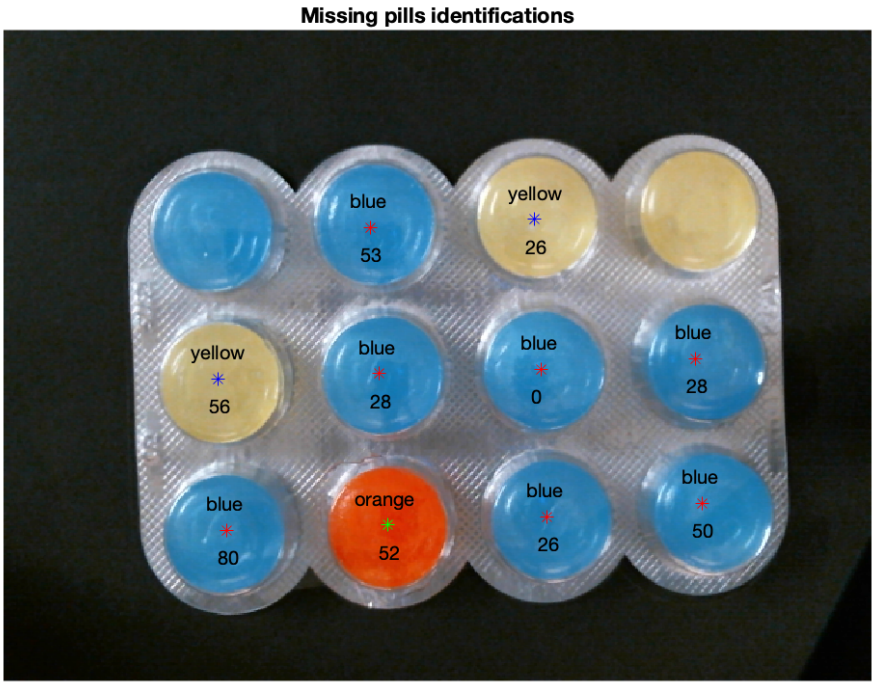

Figure 10: Results for a different input image.