FINAL POSTER

Group 05

PROCESS FLOWCHART

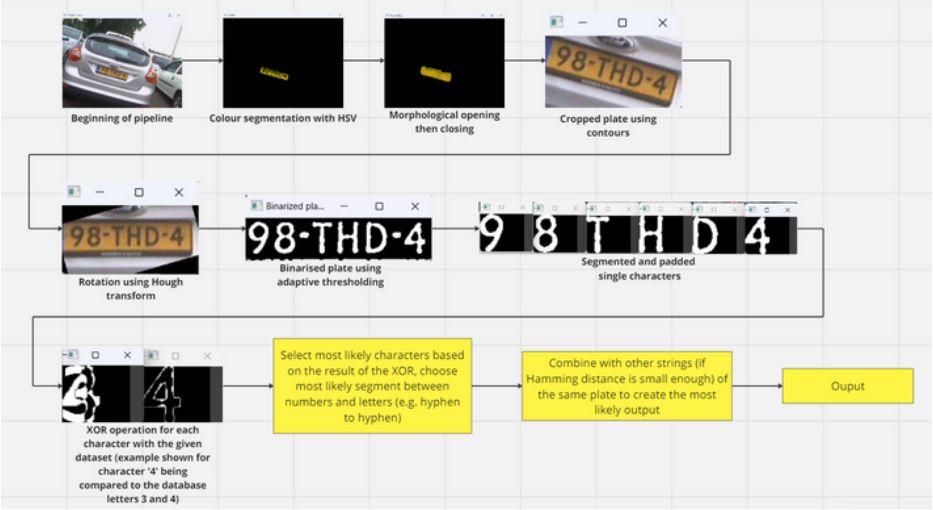

Figure 1 - Flowchart of the license plate recognition pipeline

EDGE CASES

We handled edge cases for character recognition, such as differentiating between 'S' and '5' or '8' and 'B', by grouping characters based on the hyphen positions. In fact, Dutch license plates only contain characters of a single type (numbers or letters) before, between and after the hyphens Therefore, for each segment we store the most likely characters and numbers, and select which to use based on their cumulative XOR score. Naturally, however, this did not handle all edge cases, as the 'P' and 'F' characters were also sometimes inaccurately exchanged. Furthermore, another edge case we encountered was the presence of irrelevant yellow objects in the image, which needed to be ignored. We achieved this by ensuring that each suspected license plate had an aspect ratio within a pre-defined range (the width must be between 1.8 and 4.2 times bigger than the height).

TRANSITION HANDLING

In order to determine when a new car is being shown, we use the Hamming distance of the current detected string to the combination of previous ones (the combination process is further explained in the **Merging Frames** section). If the distance is at least 4, then we conclude that a transition occurred. Additionally, due to one of the pre-defined requirements being that a plate must be visible for at least 2 seconds, we ensure that the time elapsed since the last detection is at least of the aforementioned time. This allowed us to prevent brief erroneous readings, potentially caused by a mishap within our pipeline (such as the angle of the plate not having been adjusted adequately).

MERGING FRAMES

We combine multiple detections from the same plate by maintaining an array of all the strings we believe are from the same plate (using the aforementioned Hamming distance and seconds elapsed). Thereafter, we merge the elements of the array by selecting the most chosen character for each index (regardless of hyphens), thus creating the final string which we output.

EVALUATION AND FURTHER IMPROVEMENTS

Using the provided training video, our final pipeline achieves 90%, 100% and 20% true positive rates for categories 1, 2 and 3 respectively. Due to the use of colour segmentation without any backup options, we were not able to recognise any plates in category 4. Furthermore, we had 2 false negatives for category 1 and 2 for category 3.

Unfortunately, the pipeline does not handle category 3 as well as we expected. This is due to each plate being rather small, rendering the binarisation of the image inaccurate, and thus resulting in poor character recognition. Whilst we try to compensate for this in our pipeline, it is clear that further adjustments needed to be made (although we feel that the pipeline did not need to change drastically, rather some parameters needed fine-tuning). Additionally, had we had more time, we would have attempted to tackle category 4 by localising plates through other means, potentially by finding straight lines and 90-degree edges within the image.