Question 1: Codebook creation

Question 1c: codebook visualization

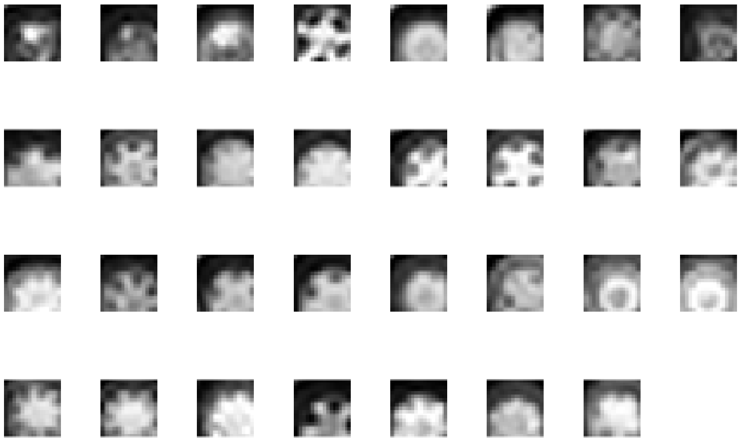

Figure 1: A selection of patches corresponding to a cluster in the codebook

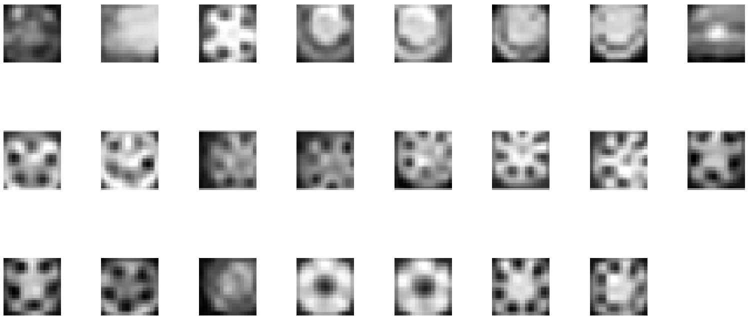

Figure 2: A selection of patches corresponding to a cluster in the codebook

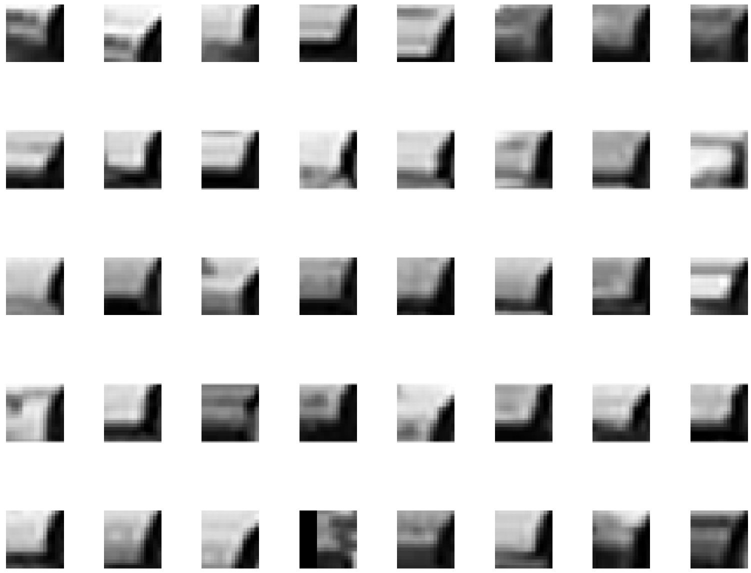

Figure 3: A selection of patches corresponding to a cluster in the codebook

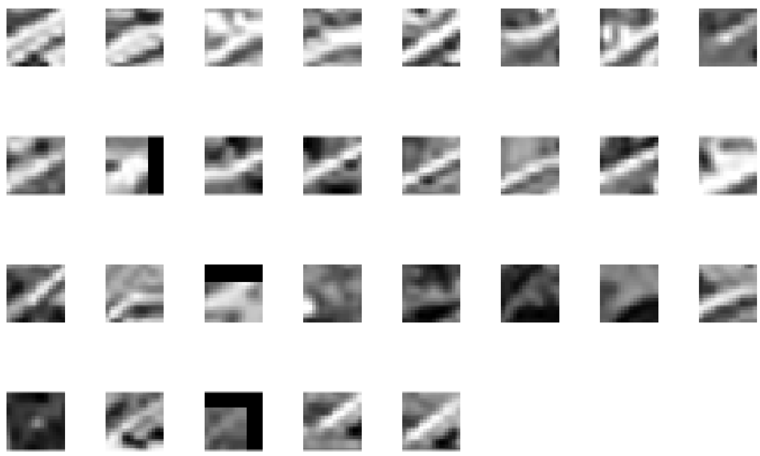

Figure 4: A selection of patches corresponding to a cluster in the codebook

Note: a maximum of forty image patches are displayed in figures 1 through 4. More occurrences could have been assigned to a particular cluster, in that case these are not displayed

Figures 1 through 4 contain various image patches corresponding to clusters in the codebook. There are distinct visual similarities in the patches within the cluster, for example the corner and arch on the right side of patches in figure 3, the diagonal line (with a similar angle) in figure 4, or the wheel spokes in most patches in figure 2.

Question 2: Occurrence generation

Question 2c: Relative occurrence location

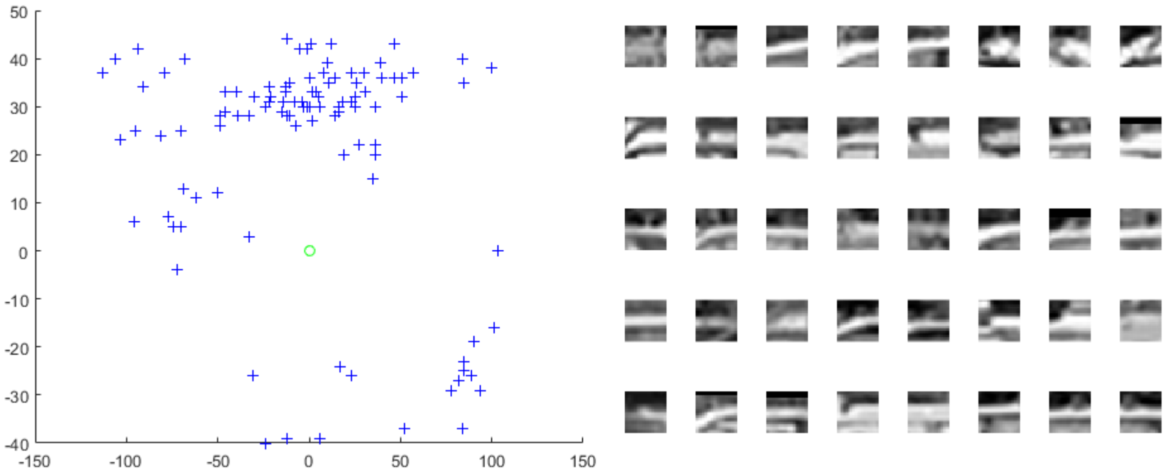

Figure 5: Occurrences locations with respect to reference point (green), and occurrence image patches

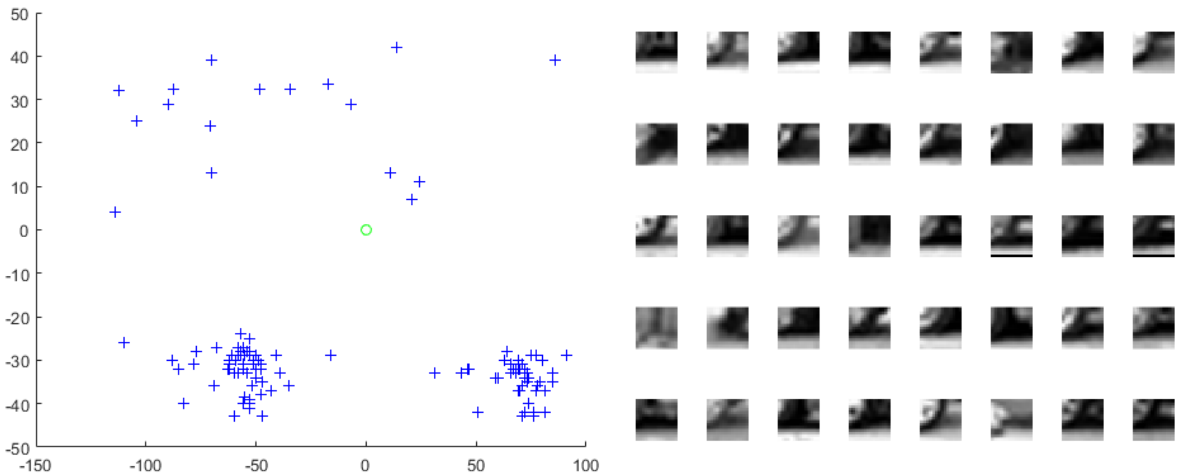

Figure 6: Occurrences locations with respect to reference point (green), and occurrence image patches

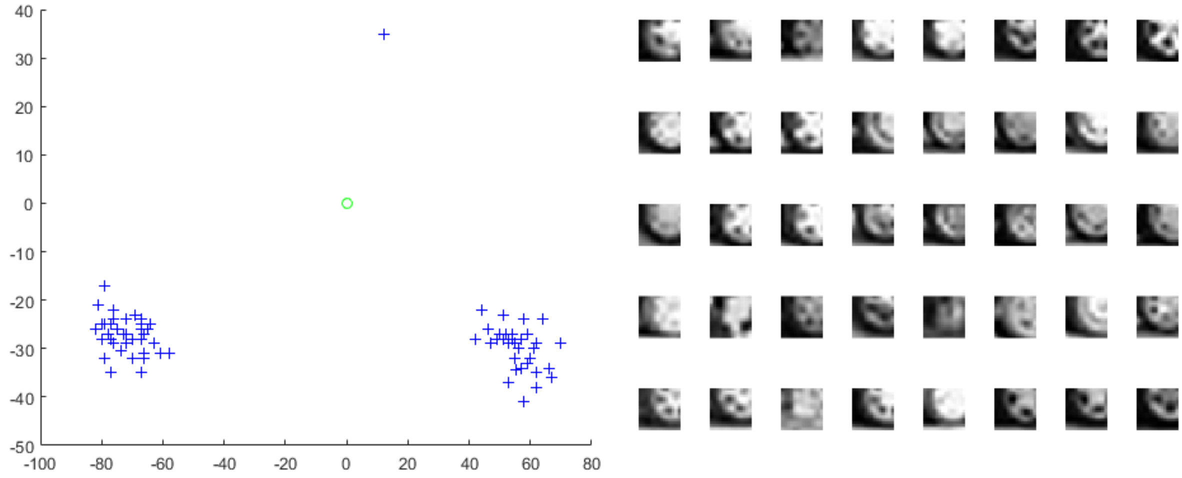

Figure 7: Occurrences locations with respect to reference point (green), and occurrence image patches

Note: a maximum of forty image patches are displayed on the right in figures 5, 6 and 7. In the relative position plot on the left, all occurrences are included. Therefore not every point in the relative position plot can have it's corresponding occurrence displayed

Overall we believe the relative positions of the occurrences roughly represent the expectations. In figure 5, we think most often the occurrence represents a part of the car roof, which corresponds to most of these patches being located above the reference point (image center, roughly car center). Some noise in the relative position distribution could be attributed to curb edges or other horizontal lines, present in other regions of the training images. In Figures 6 and 7, we believe the patches represent parts of wheels, which correspond to large localized groups of points located below and to the right and left of the reference point.

Question 3: Recognition using simple ISM

Question 3c: Results of recognition

Note: We used a threshold of 1.0 for a local maximum of the hough accumulator to be considered a detection. All such detections are displayed as a red x. Up to three detections with the highest score have bounding boxes drawn - top score in red, second in green and third in blue.

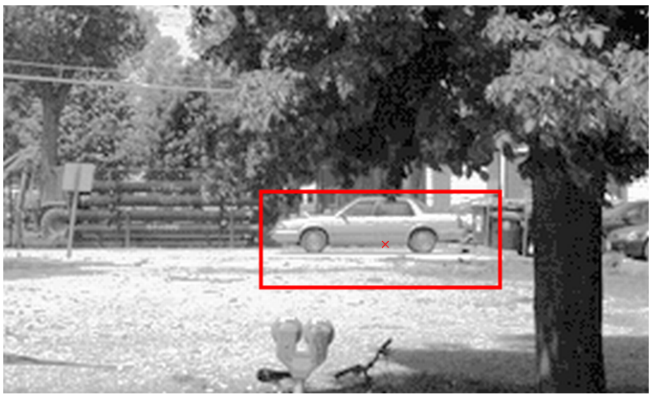

Figure 8: Success case in detection

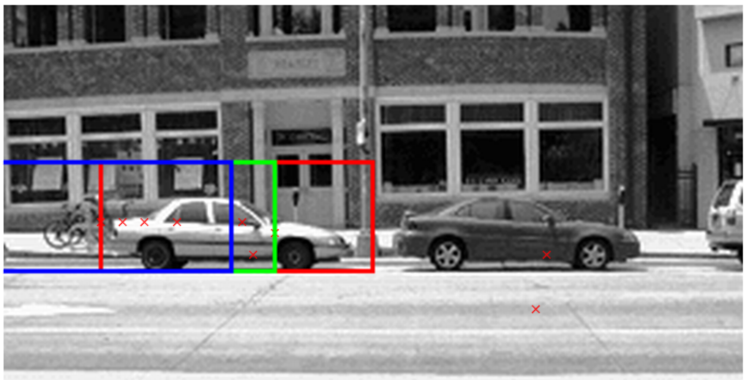

Figure 9: At least one true positive detection. False positives likely caused by bicycle wheels and background noise. Detection on second car is not in top 3 by score.

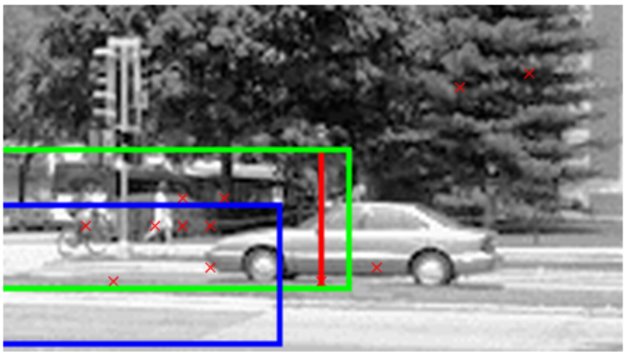

Figure 10: False positives possibly caused by bicycle wheels. Detection on car is not in top 3.

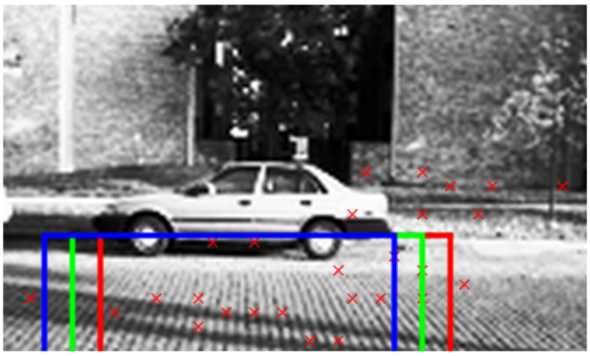

Figure 11: Many false positive detections. Possibly caused by strong texture in foreground.

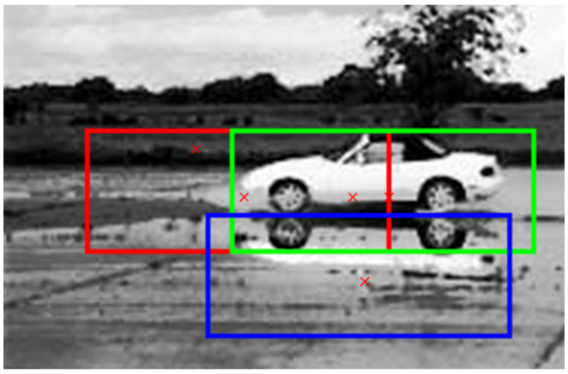

Figure 12: Top scoring detection is a false positive, but second best is a true positive. Third best is an upside down reflection, which is a likely cause for the false positive

Improvement suggestions

- In this simple ISM model, our codebook does not take the scale of the object (car) into consideration.If a part detected in a test image is on a target object, which is scaled in comparison to the training images, the votes in the Hough accumulator will spread out and true detections may have a lower scores than false positives. (See figure 13 for an illustration).
- Rotation of the car is also not considered. In case the car in the test image is rotated, the voting from the different parts will once again spread between various boxes.
- We think both of these issues could be helped by keeping information about scale and orientation in the codebook, and detecting it at time From a detected occurrence we would need to detect its scale and orientation, and adjust it's vote accordingly.

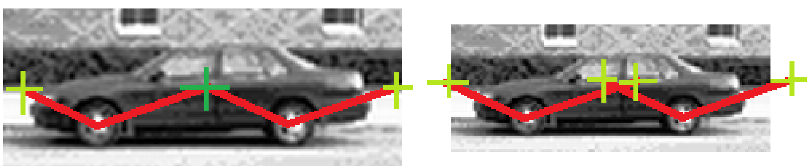

Figure 13: Illustration of issues with scaling. Similar issues arise with rotation