Line Keep Assist

By Davina Sanghera

1. Introduction

This project implements a Line Keep Assistant (LKA) using two different approaches. First is a classical computer vision pipeline that uses colour and gradient thresholding, perspective warping and polynomial curve fitting. Second is a deep learning pipeline that uses LaneNet and clusters and curve fits its outputs. Both methods aim to detect the left and right ego lanes from driving videos. The detected lanes are overlaid onto the original frames and an annotated output video comparing the classical versus deep learning approaches is generated. Evaluation metrics such as side detection accuracy, mean lateral offset and average latency are computed on the TuSimple dataset (daytime highway) and visual evaluation is performed on the CULane (urban with faded paint) dataset.

2. Methodology

2.1 Classical CV Pipeline

In the classical method, the region of interest is cropped to extract the bottom 50% of the image and isolate the road. The ROI is converted to HLS color space, and the S channel is thresholded between 120 and 255 to highlight lane paint while minimising noise. Additionally, a horizontal Sobel gradient is computed and used to threshold the L channel between 70 to 255 to emphasise vertical edges. These two binary masks are then combined.

A perspective transform is applied using four manually selected source points and the vertices of a destination rectangle, so that the mask is warped into a birds eye view. These points were fine-tuned experimentally to vertically align the lanes in the warped image. A histogram-based sliding window search method is used to extract the lane pixels, with a window height of 10 and a margin of 80 pixels. More windows and wider windows were chosen to better handle dashed lanes and curved lanes.

A second-order polynomial is fit to each lane using NumPy's polyfit, as I found that third-order polynomials often fit to noise. The confidence is computed as a weighted combination of pixel count, fit residual, and temporal consistency with the previous frame. If the confidence is higher than 0.6, the lane is marked as detected. The fitted curves are finally projected back to the original camera view using the inverse perspective transformation and they are drawn in green (left) and blue (right). If the confidence is below the threshold, the curve is drawn in grey and deemed as 'Not Detected'. Lateral offset is computed at a fixed y-lookahead of 400 pixels from

the bottom of the frame, which corresponds to approximately 20 meters ahead in real-world distance. All metrics are saved to classical_metrics.csv.

2.2 Deep Learning Pipeline

The deep learning method uses a pretrained LaneNet model to generate a binary segmentation mask of all detected lane pixels. The output mask is clustered using DBSCAN to separate individual lanes in the frame. Each cluster is then fit with a second-order polynomial using the same fitting logic as the classical method.

Since LaneNet outputs all visible lanes, ego lanes are selected by choosing the two clusters closest to the image center at the bottom of the frame. The confidence is computed from the model's segmentation probability map and the temporal consistency of the fitted curves. The lanes are overlaid the same as in the classical pipeline: curves are drawn in green or blue if detected confidently, or otherwise in grey. All metrics are saved to dl_metrics.csv.

3. ADAS Context

The line keep assist is a key ADAS feature that helps a vehicle maintain its position within a lane. It relies on accurate lane detection in order to perform steering corrections or issue driver alerts. The confidence scoring system is a critical component, since low confidence detections shouldn’t be able to trigger torque-based actions, and repeated uncertainty in the lane detections should prompt a message for the driver to take over. Temporal stability is also important, since it smooths out detection noise and prevents sudden lane jumps or flickering, which could otherwise lead to erratic corrections.

In failure cases such as shadows, faded paint or lane merges, the system defaults to the last reliable fit or flags the lane as undetected. If both lanes are undetected or if the confidence remains low over multiple frames, then the LKA system should disengage and alert the driver to assume control. The system's actions are constrained by its Operational Design Domain (ODD), which defines the conditions under which the system is expected to function safely. The ODD of the LKA typically includes daytime roads with clear weather conditions, where lane markings are clearly visible. Outside this domain, the fallback logic limits the system's actions and gives control back to the driver when needed.

4. Evaluation Metrics

The evaluation script computes five key metrics from the CSV file containing per-frame lane predictions:

- *Left/Right Detection Rates:* Percentage of frames where each lane is detected (surpassed confidence threshold).
- *Side Detection Accuracy:* Average of left and right detection rates.
- *Mean Lateral Offset:* Average distance of the vehicle centre to the lane centre at approximately 20m lookahead.
- *Lateral Offset Standard Deviation:* Variation in lateral offset across frames, lower value means smoother tracking.
- *Latency per Frame:* Average processing time per image.

5. Results

5.1 TuSimple (Highway)

Model	Left Detection Rate (%)	Right Detection Rate (%)	Side Detection Accuracy (%)	Mean Lateral Offset (m)	Lateral Offset Std. Dev (m)	Frames Evaluated	Avg. Latency (ms/frame)
Classical	92.86	99.29	96.07	0.344	0.361	140	20.80
Deep Learning	100.00	100.00	100.00	0.175	0.208	140	23.54

Figure 1: *Classical vs DL Lane Overlays for TuSimple*

5.2 CULane (Urban)

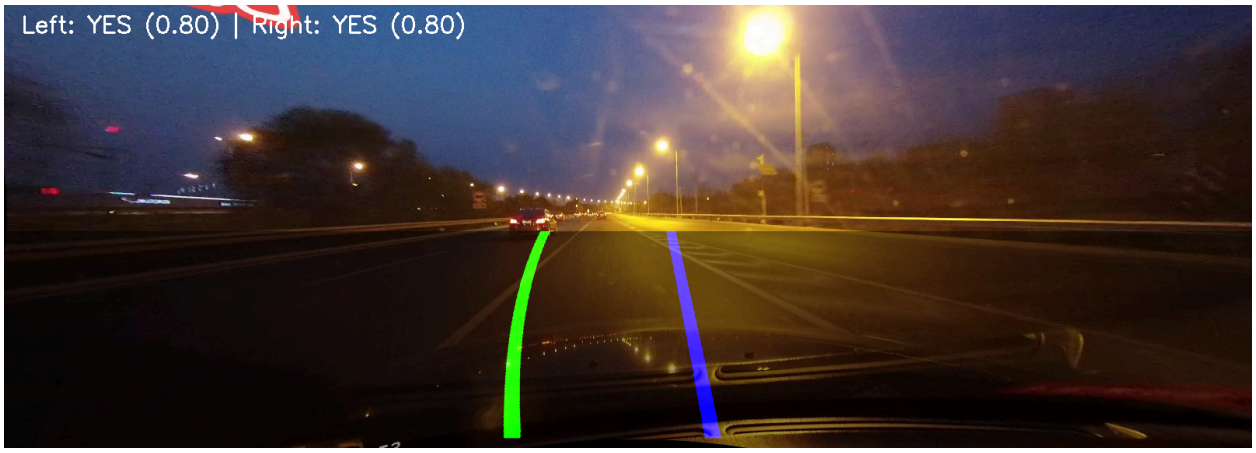

Figure 2: *Classical Lane Overlays for CULane*

6. Discussion

The results from the TuSimple table show that the deep learning model outperforms the classical method across almost all of the metrics. The DL model has perfect side detection rates and a much lower mean lateral offset (0.175 m vs 0.344 m), meaning it could track lane positions more accurately, with less deviation from the true centre line. The small increase in latency (23.54 ms vs 20.80 ms), caused by the simultaneous performance of feature extraction, detection and classification, is a reasonable trade-off for these improvements.

However, these metrics can be slightly misleading, because the DL model tends to overpredict its confidence scores, often assigning high confidence scores even when it extrapolates incorrectly and fails to account for curved sections of the road ahead (as is evident in Figure 1). This suggests that while its predictions are accurate in straight, well-lit road conditions, the model tends to be overconfident in more challenging scenes. On the other hand, although the classical model is less accurate overall, it has more realistic confidence behaviour; its confidence drops appropriately when the markings become less clear. For example, in the CULane night frames, it initially detects the lanes but then its confidence gradually drops to around 0.5 and stays below the confidence threshold, reflecting the genuine uncertainty caused by the low illumination.

Overall, the DL provides more robustness and precision in ideal conditions, but the classical approach provides more trustworthy confidence scoring. Future improvements could combine the strengths of both models by using confidence calibration or adaptive thresholding methods to adjust the probability map outputted by the DL model. Also training with additional nighttime, curved road and adverse weather data would likely improve the models generalisation.

7. Conclusion

In conclusion, the DL approach quantitatively showed clear improvements over the classical approach in terms of detection accuracy and stability on the TuSimple dataset. However, while the DL model achieved near-perfect detection rates and lower lateral offsets, it also tended to overestimate its confidence scores, especially in curved road scenarios. This highlights a trade-off between precision and reliability. Future work should focus on improving the DL model's confidence scoring and increasing its exposure to more complex environments, in aims of achieving both high accuracy and honest performance in real-world conditions.