Case Study: Design and Optimization of a Lane-Keeping Assist (LKA) System in an ADAS Control Unit

1. Problem Statement

With increasing demand for Level 2+ autonomy in passenger vehicles, OEMs require robust and fail-safe Lane-Keeping Assist (LKA) functionalities. One of our clients—a Tier 1 automotive supplier—faced challenges in:

- Maintaining high-accuracy lane detection under poor visibility (rain, night, faded markings).
- Reducing latency between lane detection and actuation.
- Integrating the LKA feature into their existing Electronic Control Unit (ECU) architecture without exceeding thermal or memory budgets.

2. Our Approach

We proposed a comprehensive co-design approach, combining embedded hardware optimization, sensor fusion, and real-time software algorithms using a model-based design (MBD) workflow.

Key Steps:

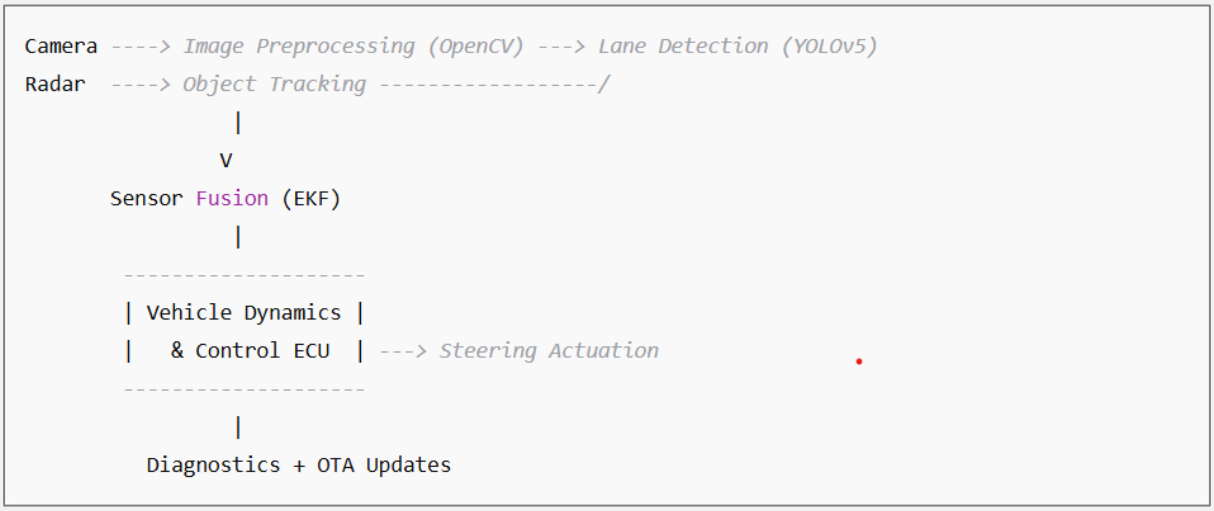
1. Requirement Analysis
    - Functional Safety: ISO 26262 compliance up to ASIL B.
    - Latency target: <50ms from perception to actuation.
    - Environment: Urban and highway, day/night, rain/light fog.
2. Architecture Co-Design
    - Proposed a dedicated Vehicle Control Domain Controller (VCDC) based on an NXP S32G or Renesas R-Car SoC.
    - Designed software using AUTOSAR Adaptive Platform for dynamic service deployment and high-performance compute.
3. Sensor Fusion Layer
    - Fused camera (RGB + IR) and radar (77GHz) inputs using an Extended Kalman Filter (EKF).
    - Integrated lane-level HD map data via a low-bandwidth V2X module.
4. Software Design & Simulation

- Developed LKA control algorithms using MATLAB/Simulink and Simulink Vehicle Dynamics Blockset.
- Simulated various edge cases with CarSim and IPG Carmaker.
- Deployed perception stack using OpenCV, TensorFlow Lite, and YOLOv5 for lane marker segmentation.

5. Hardware-in-the-Loop (HIL) Testing
   - Performed MIL/SIL testing in Simulink.
   - Executed real-time HIL on dSPACE MicroAutoBox II and NI PXI systems.

3. The Solution

System Architecture Diagram



Key Components

- ECU Platform: NXP S32G399A
- OS: QNX + AUTOSAR Adaptive
- AI Models: YOLOv5m trained on BDD100K + custom data
- Middleware: SOME/IP, DDS (for ROS2 compatibility)
- Safety Stack: ISO 26262 QM to ASIL B certification for lane-keeping module
- Comms: CAN FD, Ethernet (TSN), V2X (DSRC)

4. Outcomes & Results

Metric	Before Optimization	After Optimization
Lane Detection Accuracy (Night)	68%	91%
Perception-to-Actuation Latency	87 ms	44 ms
CPU Load (at 60 fps)	93%	63%
Memory Usage (RAM)	1.8 GB	1.1 GB
Fault Recovery Time	NA	200 ms

5. Learnings & Best Practices

a. What Worked

i. Co-simulation with Simulink + CarSim helped test rare corner cases before on-road testing.

ii. Using modular adaptive AUTOSAR stack enabled faster OTA deployment and debugging.

iii. Combining deep learning with traditional vision processing greatly improved lane marker robustness.

b. Challenges Faced

i. Model optimization for real-time execution on embedded hardware was non-trivial; required pruning and quantization.

ii. Thermal budget constrained compute; applied DVFS (Dynamic Voltage and Frequency Scaling) techniques.

6. Tools & Tech Stack Used

Category	Tools
Algorithm Design	MATLAB, Simulink, CarSim, IPG Carmaker
AI/Perception	YOLOv5, TensorFlow Lite, OpenCV
Middleware	AUTOSAR Adaptive, DDS, SOME/IP
ECU & HIL	dSPACE, NI PXI, NXP S32G
Development	C++, Python, Embedded Linux, QNX
Testing	CANoe, VectorCAST, Jenkins CI/CD

7. Conclusion

This case study demonstrates how your company's end-to-end ADAS system integration capabilities—from hardware-software co-design to perception, control, and HIL testing—can solve real-world challenges for next-gen autonomous features. By leveraging industry-standard platforms and tools, the LKA feature was delivered with higher reliability, faster response, and lower resource usage, setting a benchmark for mid-level autonomy.