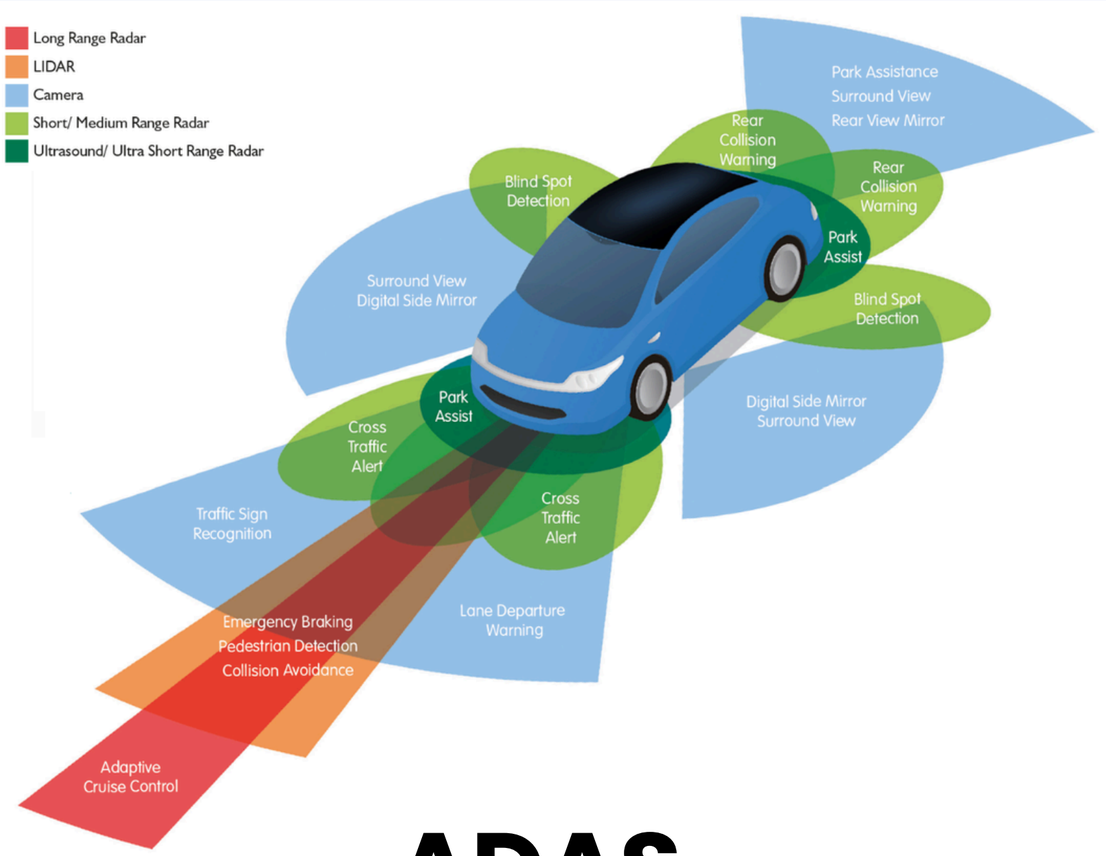
TECHNISCHE HOCHSCHULE DEGGENDORF

Camera and CAN Integration in Advanced Driver Assistance Systems

Presented By: Deep Bharatbhai Savaliya



ADAS

ADVANCED DRIVER ASSISTANCE SYSTEM

The Role of Camera Sensors and CAN Communication in ADAS

- **Camera Sensors:**
  - Function: Detect objects, lane markings, pedestrians, and traffic signs.
  - Key Properties: High resolution (1.2 MP to 8 MP), HDR, wide FOV, 25-60 fps for real-time monitoring.
  - Challenges: Environmental influences like fog, glare, and temperature affecting performance.

- **CAN Communication:**
  - Function: Real-time data transmission between ECUs (Electronic Control Units) in vehicles.
  - Key Properties: Linear bus topology, 1 Mbit/s max bitrate (CAN 2.0), supports up to 8 bytes payload (CAN 2.0).
  - Challenges: Limited bandwidth; raw image data cannot be transmitted efficiently; requires Ethernet for high-bandwidth data.

- **Integration in ADAS:**
  - Stereo Vision: Uses two cameras for distance measurement via disparity.
  - Object Data Transmission: Efficiently sent over CAN, suitable for real-time ADAS functions (e.g., braking, lane-keeping).
  - Raw Image Data: Requires Ethernet or higher bandwidth protocols due to large data size.

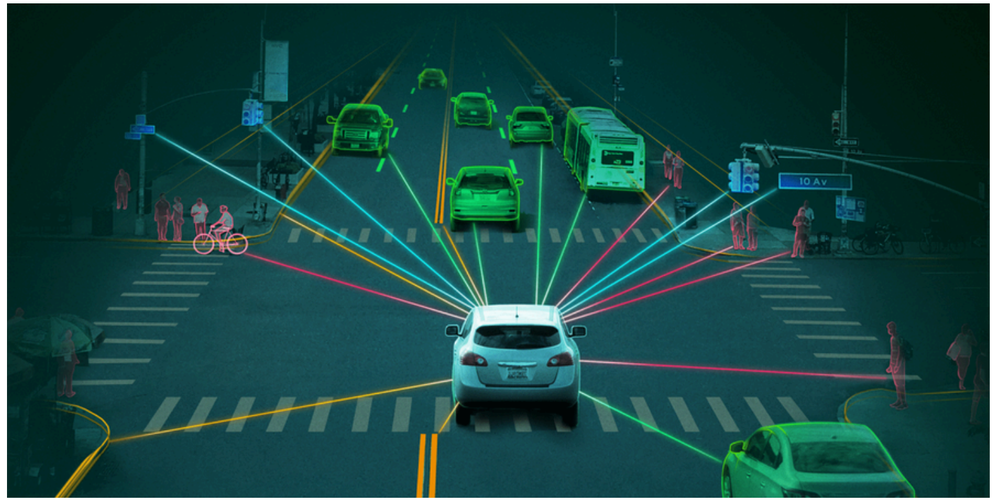
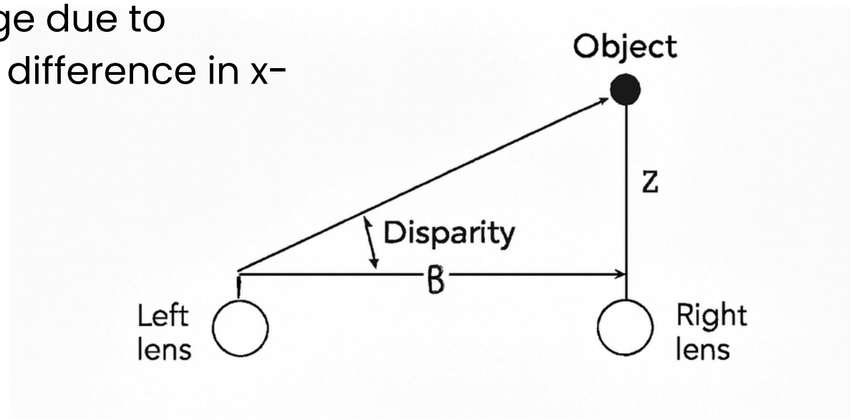


image source

Distance Measurement Using Stereo Camera – Disparity

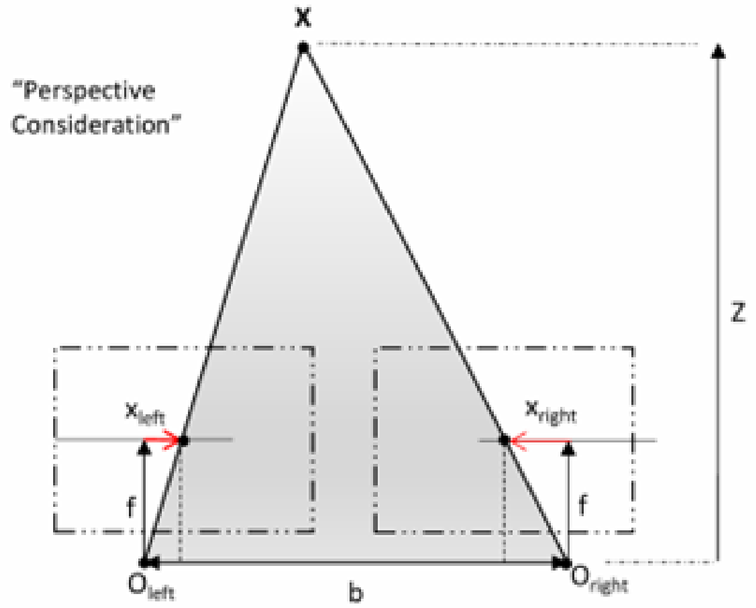
A stereo camera system uses two horizontally aligned cameras with a known baseline to capture a 3D scene. Objects appear at different x-positions in each image due to viewpoint differences, creating a horizontal disparity d, calculated as the difference in x-coordinates of matching pixels in the left and right images.



The disparity is related to the depth (Z) of the object as follows:

$$Z = \frac{b \cdot f}{d}$$

Where:
d = Disparity (pixel shift between the two images)
b = Baseline (distance between the left and right camera centers)
f = Focal length of the camera lens
Z = Depth (distance to the object)



- For example, assuming a stereo camera system with:
  - Baseline b = 12 cm = 0.12 m
  - Focal length f = 1000 pixels
  - Left image x-coordinate x_left = 560 pixels
  - Right image x-coordinate x_right = 530 pixels

d = $x_{\text{left}} - x_{\text{right}}$ = 560 – 530 = 30 pixels

$$Z = \frac{0.12 \cdot 1000}{30} = \frac{120}{30} = 4.0\ Meters$$

CAN Layer 2: Net Data Rate

- **Maximum Net Data Rate (Without Higher Protocol Layers):**
  - The net data rate at Layer 2 refers to the proportion of transmitted bits that represent the actual payload data, excluding protocol overhead (e.g., control, identifier, CRC, and ACK fields). This is crucial for embedded automotive systems, where bandwidth is constrained.
- **Calculation of Efficiency:**
  - Efficiency η of the protocol is defined as the ratio of data bits to total bits transmitted:

$$\eta = \frac{D_{payload}}{D_{total}}$$

$D_{payload}$: Number of payload bits (actual user data).

D_{total}: Number of total transmitted bits in one CAN frame (including all overhead).

- **Formula for Net Data Rate:**

$$R_{net} = \eta \, . R_{gross}$$

$$R_{net} = \frac{D_{payload}}{D_{total}} . R_{gross}$$

- **Example**

Max payload: $D_{payload}$ = 64bits (8 data bytes)

Typical full frame: $D_{total} \approx$ 130bits

So: $\eta = \frac{64}{130} \approx 0.492$

Then:

For $R_{gross} = 500kbit/s$:

$$R_{net} = 0.492 \, . 500 = 246kbit/s$$

For $R_{gross} = 1Mbit/s$:

$$R_{net} = 0.492 \, . 1000 = 492kbit/s$$

CAN Layer 2: Propagation Time

- **Bit Time Calculation:**

If 1,000,000 bits are sent in 1 second, then 1 bit takes $\frac{1}{1000000} = 1\mu s$

$$T_{bit} = \frac{1}{bitrate}$$

For bitrate = 500 kbit/s

$$T_{bit} = \frac{1}{500 \times 10^3} = 2\mu s$$

For bitrate = $1Mbit/s$

$$T_{bit} = \frac{1}{1 \times 10^6} = 1\mu s$$

- **Propagation Delay:**

The propagation delay Tprop is the time it takes for a signal to travel through the medium (e.g., copper wire), given by:

$$T_{prop} = \frac{L}{v}$$

L: Length of the cable (meters)
v: Propagation speed of the signal (meters/second)

Example: For L= 20m, V=0.66.3 × $10^8 m/s$

$$T_{prop} = \frac{20}{0.66 \times 3 \times 10^8} \approx 0.101\mu s$$

- **Arbitration Requirements:**
  - To maintain proper arbitration on the CAN bus, the propagation delay must be significantly smaller than the bit time:

$$T_{prop} < 0.1 . T_{bit}$$

Signal Scaling in CAN Communication

Signal Scaling Overview:

- In automotive communication systems like CAN, raw binary values need to be converted into physical (interpretable) values. This process involves applying a scaling factor and offset to the encoded binary signal.

- **Physical Value Calculation:**

  A signal's physical value is mapped to a defined range using the scaling factor SK and the offset SO:

  Physical Value = Raw Value × SK + SO

  Where:

  - Raw Value: The binary decoded value from the CAN frame
  - Scaling Factor (SK): The size of each interval, i.e., how much physical quantity one bit step represents
  - Offset (SO): The minimum physical value corresponding to a raw value of zero

Derivation of Scaling Factor and Offset:

Given a signal encoded using n bits, the physical range is defined between Physical Min and Physical Max. The scaling factor SK is calculated as:

$$\text{Scaling Factor (SK)} = \frac{Physical\ Max - Physical\ Min}{2^{Length} - 1}$$

Example: If the signal range is from 0 to 10 meters with 4 bits:

Bit length: 4 bits

Physical range: 0 to 10 meters

Offset (SO): 0

$$SK = \frac{10 - 0}{2^4 - 1} = \frac{10}{15} \approx 0.666667$$

PDU Mapping in CAN Communication

- **Protocol Data Unit (PDU) Layout:**
  - In CAN, signals are encoded in a PDU, which contains both control and data fields. The PDU must be correctly mapped to ensure that each signal occupies its designated bit positions in the frame, preventing errors.
- **Bit-Level PDU Mapping:**
  - The layout of a 64-bit PDU frame shows how each signal is assigned specific bit positions. The diagram visualizes the mapping of various signals to their respective bit fields within the 64-bit CAN frame.
  - The horizontal axis represents bit positions (0–63), and each signal is assigned to specific positions based on its length.

Example: If the signal range is from 0 to 10 meters with 4 bits:

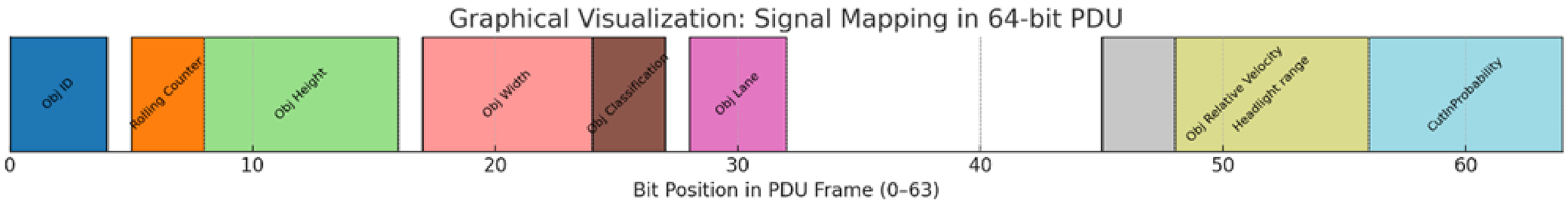
Bit length: 4 bits

Physical range: 0 to 10 meters

$$SK = \frac{10-0}{2^4-1} = \frac{10}{15} \approx 0.666667$$

Offset (SO): 0

- **Visualizing the Mapping:**
  - The PDU frame is divided into discrete bit fields, each representing a specific signal, with boundaries between bytes clearly marked. The mapping ensures that each signal is correctly interpreted by receivers.



Bit-Level PDU Mapping (Visualization)

Signal	Length (Bits)	Physical Range	Formula for SK	SK (Scaling Factor)	SO (Offset)	Unit
Obj ID	4	0 to 10	$\frac{10-0}{2^4-1}=\frac{10}{15}$	0.666667	0	-
Rolling Counter	3	0 to 8	$\frac{8-0}{2^3-1}=\frac{8}{7}$	1.142857	0	-
Obj Height	8	0 to 10	$\frac{10-0}{2^{10}-1}=\frac{10}{255}$	0.039216	0	m
Obj Width	7	0 to 5	$\frac{5-0}{2^7-1}=\frac{5}{127}$	0.039216	0	m
Obj Classification	3	0 to 7	$\frac{7-0}{2^3-1}=\frac{7}{7}$	1.0	0	-
Obj Lane	4	0 to 8	$\frac{8-0}{2^4-1}=\frac{8}{15}$	0.533333	0	-
Obj Relative Velocity	11	-35.0 to +35.0	$\frac{35-(-35)}{2^{11}-1}=\frac{70}{2047}$	0.034203	-35.0	m/s
Headlight Range	8	0 to 250	$\frac{250-0}{2^8-1}=\frac{250}{255}$	0.980392	0	m
CutInProbability	8	0 to 100	$\frac{100-0}{2^8-1}=\frac{100}{255}$	0.392157	0	%

Feasibility of Transmitting Camera Sensor Data over CAN Layer 2

Data Type	Description	CAN Layer 2 Feasibility
Object Data	Processed data like object ID, position, velocity, classification, and lane index.	Feasible: Object data is small, structured, and well-suited for real-time transmission over CAN. It fits within the 8-byte frame size, allowing for efficient transmission.
Raw Data	Unprocessed data such as full-resolution images or point clouds.	Not Feasible: Raw data, especially high-resolution images, require much more bandwidth than CAN can provide. Even a single image frame would take too long to transmit, making real-time communication impractical.

Why CAN Can Handle Object Data:

- Compact and Structured: Object data is small in size, easily fitting within the 8-byte maximum payload of CAN frames. It includes processed outputs like object positions, velocity, and lane information, all of which can be efficiently encoded into a few bytes.
- Real-Time Suitability: Due to its small size and structured nature, object data is ideal for real-time transmission over CAN, supporting applications such as steering and braking in ADAS.

Why CAN Cannot Handle Raw Data:

- High Bandwidth Requirement: Raw data from cameras, such as full image frames, are much larger in size. A high-resolution image requires significantly more bandwidth than CAN's maximum throughput, making it unsuitable for real-time transmission.
- Data Overflow: Transmitting raw data would exceed CAN's limitations in terms of frame size and data rate, necessitating alternative solutions like Ethernet or edge computing for processing.

Application of Edge Detection Filter to Grayscale Image

- **Edge Detection Using a Filter:**
  - Edge detection highlights areas of sharp intensity changes in an image, helping to identify boundaries between objects or regions.
  - In this case, a 3x3 edge detection filter is applied to a grayscale image to identify edges by detecting horizontal and vertical transitions.

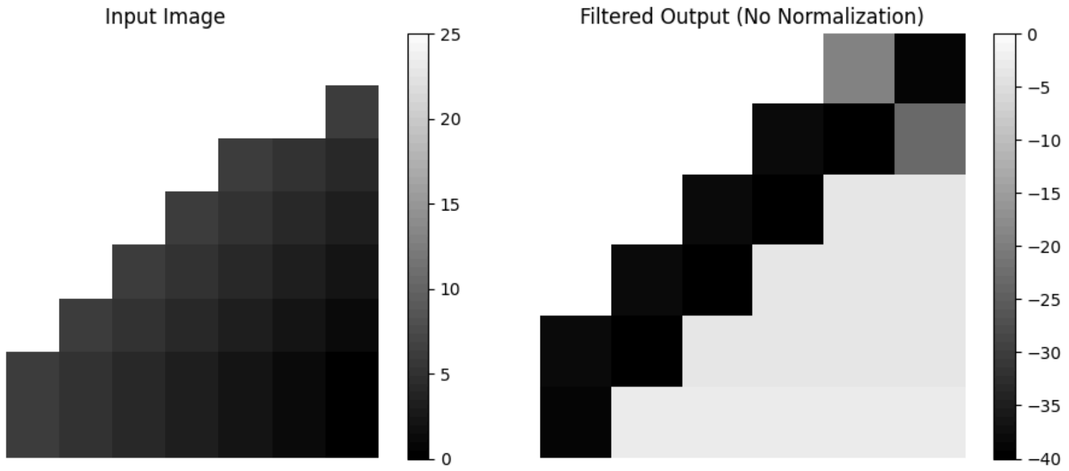
- **Filter Used:**
  - The filter kernel used is a simple Sobel-like operator designed to detect edges in the horizontal and vertical directions:

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

  - This kernel performs a convolution operation with the input image to detect intensity gradients.

- **Resulting Image:**
  - The original grayscale image is compared to the filtered output, showing the edges clearly.
  - The filtered output highlights areas of high contrast, which are typically edges in the image.

- **Visual Output:**



Connecting an Analysis Tool to the CAN Bus

- **Purpose of CAN Bus Analysis:**
  - Analyzing CAN bus traffic is crucial for monitoring and debugging communication between Electronic Control Units (ECUs) in an automotive network.
  - Analysis tools allow for passive monitoring, decoding, and visualizing of the data transmitted across the CAN network.

- **Typical Connection Setup:**
  - An analysis tool (e.g., Vector CANoe, PCAN-View) is connected to the CAN bus via a USB-to-CAN interface.
  - Common Interfaces: PEAK, Kvaser, or Vector adapters are used to link the tool with the CAN_H and CAN_L lines of the bus.

- **How It Works:**
  - The adapter connects to the OBD-II diagnostic port, a breakout connector, or test points on the wiring harness.
  - The tool functions as a passive listener, meaning it does not interfere with normal CAN communication. It only reads the data and decodes it using the DBC file (Database file).

- **Software and Decoding:**
  - The analysis software applies the DBC file to interpret the data in the CAN messages, presenting human-readable information such as signal names, values, and timestamps.
  - This aids in debugging, performance analysis, and reverse engineering of CAN communication for development or troubleshooting purposes.

- **Applications:**
  - Useful in automotive diagnostics, CAN bus traffic monitoring, and for understanding communication patterns within complex ADAS systems.

Multi-Modal 3D Object Detection in Autonomous Driving

- **Objective:** To improve the accuracy and robustness of 3D object detection by combining multiple sensor modalities (LiDAR, cameras).
- **Approach:**
  - Feature Representation: Combining raw data (point clouds, images) into a unified or raw representation for fusion.
  - Alignment Methods: Aligning heterogeneous data sources using projection matrices or attention mechanisms.
  - Fusion Techniques: Merging features from multiple sensors to improve detection performance.

- **Diagram:** Flowchart showing the steps from sensor data acquisition to fusion.

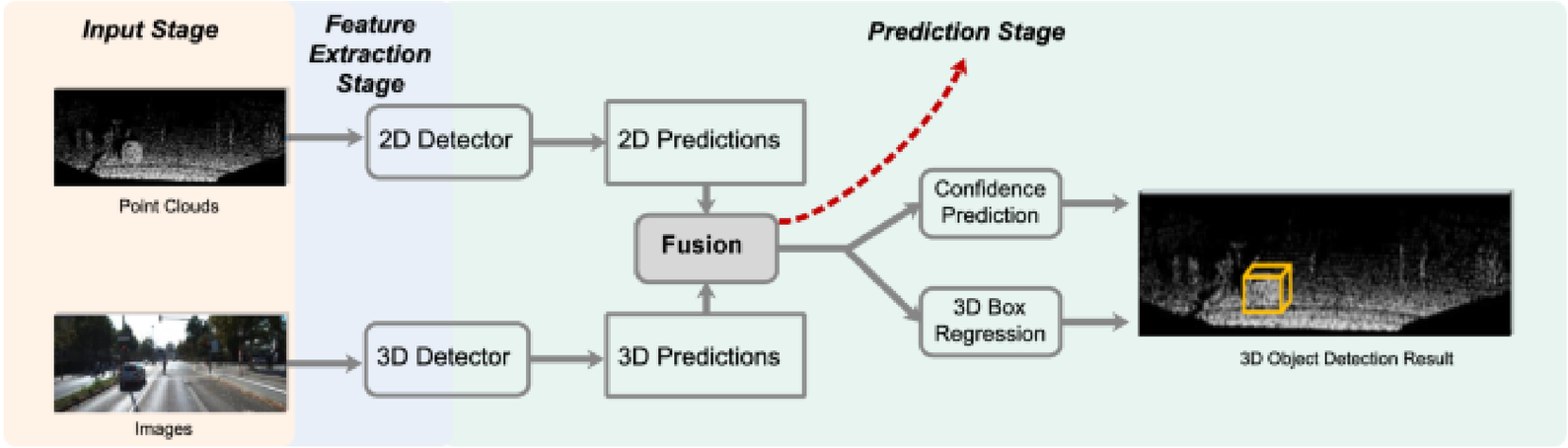


image source

Taxonomy of Multi-Modal Fusion Methods

- **Representation:**
  - Unified Representation: Converts heterogeneous data into a unified format (e.g., 2D projection of 3D point clouds).
  - Raw Representation: Uses original sensor data without pre-processing for detection tasks.
- **Alignment:**
  - Projection-Based: Uses camera projection matrices to align 3D point clouds with images.
  - Model-Based: Employs deep learning techniques like cross-attention to achieve alignment between features.
- **Fusion:**
  - Learning-Agnostic: Uses simple operations (mean, summation) to combine features.
  - Learning-Based: Uses attention mechanisms to intelligently fuse features, allowing for better performance in complex environments.
- **Diagram:** Flowchart showing the steps from sensor data acquisition to fusion.

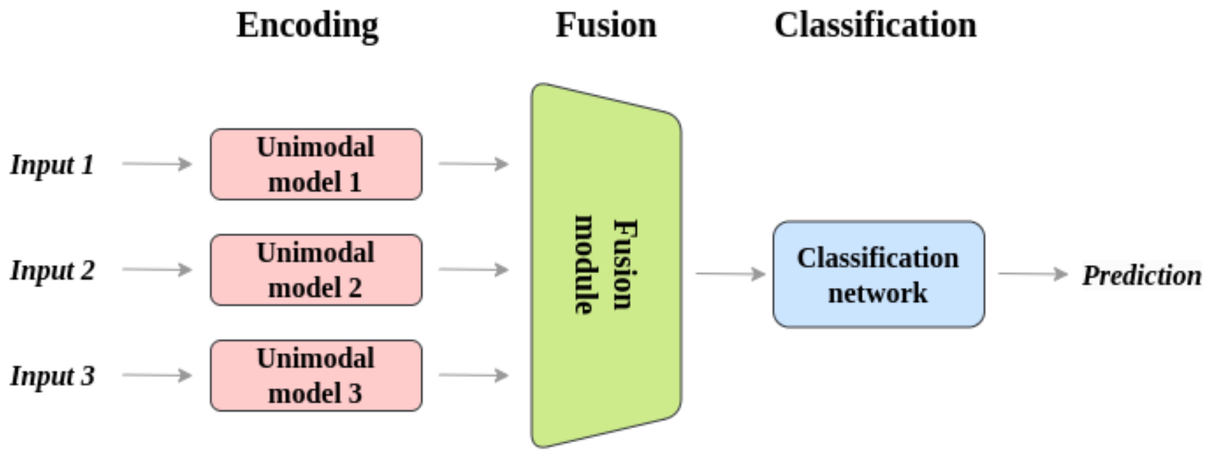


image source

Challenges and Future Trends in Multi-Modal 3D Detection

- **Challenges:**
  - Data Noise: Misalignment and inconsistency between data from different modalities lead to noise.
  - Limited Sensor Coverage: Incomplete data coverage limits detection accuracy.
  - Computational Complexity: Multi-modal fusion demands high computation power, affecting real-time performance.
- **Future Trends:**
  - Cross-Modal Data Augmentation: Synchronizing enhancements across different sensor types.
  - Temporal Synchronization: Ensuring data from different sensors is aligned in time to avoid errors in detection.
  - Improved Fusion Models: Leveraging advanced deep learning-based fusion methods for more accurate and efficient 3D detection.

Summary of Case Study

This study examined the integration of camera systems in vehicles from three key perspectives: sensing, processing, and data transmission. It was shown that object data can be effectively compressed and transmitted over CAN, whereas raw image data exceeds bandwidth limits. High-resolution sensors such as the OX08B40 offer significant advantages in detection quality. Multi-modal fusion further improves accuracy but introduces computational complexity. Future ADAS systems should leverage Ethernet for high-volume data, deploy edge AI for real-time processing, and utilize advanced fusion techniques (e.g., transformers) to support scalable and intelligent vehicle perception.

image source

image source

Thank You

Name: Deep Bharatbhai Savaliya

Matriculation No. : 12501180

Course: Master in Automotive Software Engineering

Semester: 1