THESIS

PERCEPTION ARCHITECTURE EXPLORATION FOR AUTOMOTIVE CYBER-PHYSICAL SYSTEMS

Submitted by

Joydeep Dey

Department of Electrical and Computer Engineering

In partial fulfillment of the requirements

For the Degree of Master of Science

Colorado State University

Fort Collins, Colorado

Spring 2022

Master's Committee:

Advisor: Sudeep Pasricha

Anura Jayasumana
Bret Windom

Copyright by Joydeep Dey 2022

All Rights Reserved

ABSTRACT

PERCEPTION ARCHITECTURE EXPLORATION FOR

AUTOMOTIVE CYBER-PHYSICAL SYSTEMS

In emerging autonomous and semi-autonomous vehicles, accurate environmental perception by automotive cyber physical platforms are critical for achieving safety and driving performance goals. An efficient perception solution capable of high fidelity environment modeling can improve Advanced Driver Assistance System (ADAS) performance and reduce the number of lives lost to traffic accidents as a result of human driving errors. Enabling robust perception for vehicles with ADAS requires solving multiple complex problems related to the selection and placement of sensors, object detection, and sensor fusion. Current methods address these problems in isolation, which leads to inefficient solutions. For instance, there is an inherent accuracy versus latency trade-off between one stage and two stage object detectors which makes selecting an enhanced object detector from a diverse range of choices difficult. Further, even if a perception architecture was equipped with an ideal object detector performing high accuracy and low latency inference, the relative position and orientation of selected sensors (e.g., cameras, radars, lidars) determine whether static or dynamic targets are inside the field of view of each sensor or in the combined field of view of the sensor configuration. If the combined field of view is too small or contains redundant overlap between individual sensors, important events and obstacles can go undetected. Conversely, if the combined field of view is too large, the number of false positive detections will be high in real time and appropriate sensor fusion algorithms are required for filtering. Sensor fusion algorithms also enable tracking of non-ego vehicles in situations where traffic is highly dynamic or there are many obstacles on the road. Position and velocity estimation using sensor fusion algorithms have a lower margin for error when trajectories of other vehicles in traffic are in the vicinity of the ego vehicle, as incorrect measurement can cause accidents. Due to the various complex inter-dependencies between design decisions, constraints and optimization goals a framework capable of synthesizing perception solutions for automotive cyber physical platforms is not trivial.

We present a novel perception architecture exploration framework for automotive cyber-physical platforms capable of global co-optimization of deep learning and sensing infrastructure. The framework is capable of exploring the synthesis of heterogeneous sensor configurations towards achieving vehicle autonomy goals. As our first contribution, we propose a novel optimization framework called VESPA that explores the design space of sensor placement locations and orientations to find the optimal sensor configuration for a vehicle. We demonstrate how our framework can obtain optimal sensor configurations for heterogeneous sensors deployed across two contemporary real vehicles. We then utilize VESPA to create a comprehensive perception architecture synthesis framework called PASTA. This framework enables robust perception for vehicles with ADAS requiring solutions to multiple complex problems related not only to the selection and placement of sensors but also object detection, and sensor fusion as well. Experimental results with the Audi-TT and BMW Minicooper vehicles show how PASTA can intelligently traverse the perception design space to find robust, vehicle-specific solutions.

ACKNOWLEDGEMENTS

I would like to thank all the individuals whose encouragement and support have made the completion of this thesis possible.

I would like to express my sincere gratitude to my advisor, Dr. Sudeep Pasricha, who has guided me through the years of research during my Master's Thesis. I am very thankful for the expert advice and feedback in the research process which has taught me how to explore challenging problems. Under his tutelage, I enjoyed learning a wide range of topics in Computer Architecture, Machine Learning, and Embedded Systems. His patience and mentorship towards both my research and my personal development enabled me to navigate the various intricacies of graduate school life. I am very thankful for all the project opportunities and teaching assistant positions he has offered me, which have helped me to deepen my knowledge in the domains of connected and autonomous vehicles, deep learning and embedded systems. All the help and guidance I received from Dr. Pasricha has inspired me to be a better and more responsible researcher and engineer.

I would like to take this opportunity to thank the respected members of my Master's Thesis committee, Dr. Anura Jayasumana, and Dr. Bret Windom for their valuable time and feedback.

This list cannot be complete without mentioning the company and the help from my current and former lab mates in Dr. Pasricha's EPIC lab: Vipin Kumar Kukkala, Saideep Tiku, Ninad Hogade, Shoumik Maiti, Febin Sunny, Asif Anwar Baig Mirza, and Kamil Khan.

I am blessed to have a wonderful family who have supported me throughout my years and made me the person I am today. To my loving father and mother Jayanta Dey and Supriya Dey, I draw courage from the teachings you have imparted to take on challenges in life.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF RESEARCH PUBLICATIONS

JOURNAL PUBLICATIONS:

- Joydeep Dey, Wes Taylor and Sudeep Pasricha, "VESPA: A Framework for Optimizing Heterogeneous Sensor Placement and Orientation for Autonomous Vehicles,", *IEEE Consumer Electronics Magazine*, vol. 10, no. 2, pp. 16-26, 2021

CONFERENCE PUBLICATIONS:

- Joydeep Dey, Sudeep Pasricha, " PASTA: Perception architecture search technique for Advanced Driver Assistance Systems", in *Design Automation Conference*, 2022 [SUBMITTED]

1. INTRODUCTION

This chapter outlines the challenges faced in state-of-the-art perception architecture design for automotive cyber physical platforms. The fundamental challenges observed at different levels of abstraction pertaining to raw sensor data, sensor placement, obstacle detection and sensor fusion are requisites for reliable perception architecture design and high fidelity environment modelling enabling safe navigation. This chapter also presents a general overview of the contributions of this dissertation.

1.1 MOTIVATION FOR PERCEPTION ARCHITECTURE DESIGN

The continuous evolution of technology in the domain of autonomous driving has led to the creation of Advanced Driver Assistance Systems (ADAS) equipped with automation capabilities which have increasingly reduced the requirement for driver control or intervention for mobility. By eliminating the need for human driver intervention through automation, ADAS are becoming a critical component in modern vehicles, to help save lives, improve fuel efficiency and enhance driving comfort.

The United States Department of Transportation (USDOT) projected that the increased use of driverless vehicles will cause the number of traffic deaths to reduce by about 90%. It was reported that in 2020, an estimated 38,680 people died in motor vehicle traffic crashes in the United States, representing an estimated increase of about 7.2 percent as compared to 2019 [1]. According to the report, in 2020 a 90% reduction could have saved 34,812 lives in a single year. However, in the first 6 months of 2021, an estimated 20,160 people died in motor vehicle crashes showing an 18.4% increase compared 2020, which is the *largest number of projected fatalities* in that time period since 2006 [2]. *This threat to human lives warrants the need for safer mobility solutions.*

With the benefit of highly coordinated traffic flow resulting from use of ADAS, the reduced number of accidents could reduce road *congestion*, since traffic accidents contribute to 25% of road congestions [3]. This reduction in congestion will have a direct effect on reducing vehicle related *CO_2 emission* by 60% [4]. Autonomous vehicle technology can improve *fuel economy* by at least 4% through smoother acceleration and deceleration as compared to human controlled actuation. Improvements like *reducing headway distance* between vehicles can increase *road capacity* and show a maximum improvement of fuel economy up to 10% [5]. Further, according to a report by Klynveld Peat Marwick Goerdeler (KPMG) a fully driverless mobility solution for all vehicle models can potentially reduce *travel time* by up to 40%, allowing a recovery of a total of 80 billion hours lost to commute and reduce congestion and *fuel consumption* by up to 40% [6]. Finally, Autonomous Vehicles (AVs) can provide an economically feasible solution to the *last mile problem* in a journey where consumers struggle to find connectivity between the last public transport drop-off point and their destination [7].

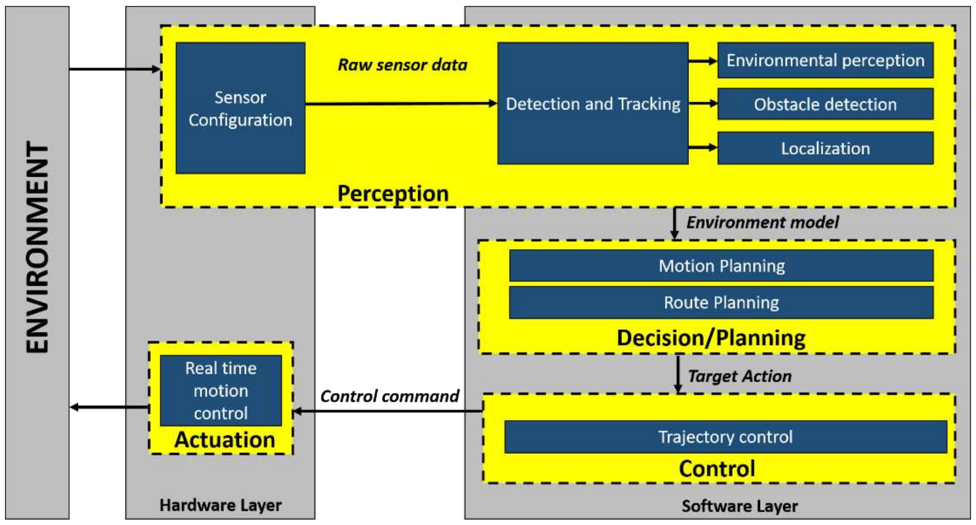


Figure 1: Overview of main ADAS modules

As shown in figure 1 above, ADAS systems are typically composed of a 4-stage pipeline involving sequential execution of functions related to *perception*, decision, control and actuation. An incorrect understanding of the environment by the perception system can make the entire system prone to erroneous decision making, which can result in accidents due to imprecise real-time control and actuation. *This motivates the need for a reliable perception architecture that can mitigate errors at the source of the pipeline and improve safety in emerging semi-autonomous vehicles.* The research presented in this thesis focuses solely on design exploration analysis of perception architectures for ADAS.

1.2 RELIABILITY CHALLENGES IN PERCEPTION ARCHITECTURE DESIGN

The capabilities of a perception architecture for a vehicle depend on the Society of Automotive Engineers (SAE) autonomy level (defined by the SAE-J3016 standard) supported by the vehicle. As developments in the automotive industry progress towards full autonomy, the role of *embedded vision* in ADAS has become critical. Currently, Original Equipment Manufacturers (OEMs) often serve as sensor vendors to automotive companies and sell products which are more suited for partial or conditional automation [8]. This poses a challenge to ADAS engineers by increasing the requirement of design of more complex and optimized vision solutions in the software layer.

In general, an optimal vehicle perception architecture should consist of carefully defined location and orientation of each sensor selected from a heterogeneous suite of sensors (e.g., cameras, radars) to *maximize environmental coverage* in the combined field of view obtained from the sensors. In addition to ensuring accurate sensing via appropriate sensor placement, a *high object detection rate* and *low false positive detection rate* needs to be maintained using efficient deep learning-based object detection and sensor fusion techniques [9]. State-of-the-art deep learning based object detection models are built with different network architectures, uncertainty modeling approaches and test datasets over a wide range of evaluation metrics. For real time perception, object detectors are resource-constrained by *latency requirements*, *onboard memory capacity* and *computational complexity* [10]. Optimizations performed to meet any one of these constraints often results in a trade-off with the performance of others. As a result, comparison and selection from among the best set of deep learning based object detectors for perception applications remains a challenge. In real-world driving scenarios, the position of obstacles and traffic are highly dynamic, so after detection of an object, tracking is necessary to predict its new position. Due to *noise* from various sources there is an *inherent uncertainty* associated with the measured position and velocity [11]. This uncertainty is minimized by using sensor fusion algorithms. An important challenge with sensor fusion algorithms is that the *complexity of tracking objects* increases as the objects get closer, due to a much lower margin for error (uncertainty) in the vicinity of the vehicle. Some of the largest remaining technical challenges for autonomous vehicle development are related to *testing conditions* and the ability of the prototype to adapt to new scenarios [12]. Autonomous vehicle testing is still limited to relatively small well-mapped environments because *generalization* for new scenarios remains a challenge. In order to generate a diverse set of controlled test conditions, developers would need to gather more data by conducting drive cycles in new environment. In the following sub-sections, we briefly discuss the significant design constraints and factors that limit the reliable performance of perception architecture design.

1.2.1 SENSOR PLACEMENT IN ENVIRONMENT PERCEPTION FOR ADAS

An important challenge facing emerging vehicles is to determine a sensor configuration that can be responsible for environment perception as per the SAE autonomy level supported by the vehicle. An optimal sensor configuration should consist of carefully selected location and orientation of each sensor in a heterogeneous suite of sensors, to maximize coverage from the combined field of view obtained from the sensors, and also maintain a high object detection rate.

1.2.1.1 TRADE-OFFS BETWEEN ADAS SENSOR MODALITIES

Table 1 depicts the trade-offs between different ADAS sensor modalities.

Table 1: Trade-offs between different ADAS sensor modalities

● : Low ●● : Medium ●●● : High

Sensor characteristics		Camera	Radar	Lidar
Distance	Range	●●	●●●	●●●
	Resolution	●●	●●●	●●
Angle	Range	●●●	●●	●●●
	Resolution	●●●	●	●●
Classification	Velocity resolution	●	●●●	●●
	Object categorization	●●●	●	●●
Environment	Low light	●	●●●	●●●
	Rain/snow	●	●●●	●●

The best-recognized ADAS sensor is the radar that has been widely adopted for position and velocity measurement. Radar is an acronym for "radio detection and ranging". It is a well-established sensor modality that detects objects using echolocation. Measurement using echolocation involves quantifying the time it takes for transmitted radio waves to reflect from a target objects. Radar was first developed by several nations for military use in the Second World War, but today it has many applications in ADAS. Since radar has been widely used in automotive systems, prototypes are well developed and relatively affordable making it attractive for car manufacturers to include in their sensor configurations [13]. For ADAS applications, radar can be divided into three categories: short-range radar (SRR), mid-range radar (MRR) and long-range radar (LRR). SRR systems traditionally used electromagnetic waves in the microwave spectrum of 24 GHz but recently there has been an industry shift towards 77 GHz due to the limited bandwidth and changing regulatory requirements. SRRs have a useful range of around 10 meters to 30 meters allowing them to support autonomous features such as park assist and lane keep assist. In ADAS, MRR and LRR are targeted for high resolution measurement using a 77 GHz frequency enabling greater accuracy for velocity and distance measurements. MRR systems operate between 30 meters and 80 meters, while LRR systems have a range extending up to 200 meters. LRRs are suitable for adaptive cruise control and forward collision warning. One of LRR's disadvantages is that it's angular resolution decreases with range so some features such as adaptive cruise control need to combine inputs from both SRR and LRR sensors having better angular resolution [13]. Radars are widely used in ADAS due to their ability to function effectively in poor weather, such as rain, snow and fog, and at night. However, it also has a limited field of view in automotive applications requiring more number of radars per sensor configuration design.

Ultrasonic sensors use echolocation from sound waves to calculate the separation distance between objects. Ultrasonic sensors have a relatively short effective operating range of around 2 meters. They are typically used in features where low driving speeds are common. Ultrasonic sensors are cost-effective and relatively robust since they are unaffected by challenging light conditions [15]. Given the limited range of established ultrasonic sensors, some manufacturers prefer using short range radars (SRR).

Lidar, a contraction of 'laser' and 'radar' is an acronym for 'light detection and ranging'. It works on the same principle as radar but uses lasers to generate a high-resolution 3D image of the surrounding environment instead of electromagnetic waves in radars. Lidar was first developed in the 1960s for meteorological, surveying, and mapping use cases, but has more recently been adopted for ADAS and autonomous vehicle development applications [16]. With the exception of Tesla, it is a widely used sensor in ADAS perception development. There are two basic types of lidars [17]. The first type uses a laser which emits pulses onto a rotating mirror which diverges the laser beam in desired directions, allowing a detection range of 300 meters or more. These are well suited for 360 degree field of view if roof-mounted. This lidar type is also compact and uses microelectromechanical systems (MEMS) technology-based rotation of mirrors to enable laser beam divergence [17]. The second type is solid-state lidars which uses an optical phased array to propagate the beams in desired multiple directions. Lidars work well in rain, snow and poor lighting conditions although it's functioning can be quite unpredictable in fog conditions [17].

Camera-based solutions have gained traction as the ADAS developer's sensor technology of choice. Cameras are also extremely cost-effective, which makes them attractive to vehicle manufacturers. Cameras used in ADAS are both monocular and stereo [18]. Forward-facing monocular camera systems are used in medium to long-range functions such as lane-keeping assistance and traffic sign detection. Rear-facing cameras serve primarily as a reversing aid for the driver, where a 2D mirror-image view of the area behind the car is displayed on a dashboard-mounted screen [19]. Forward-facing stereo cameras are more commonly used in pairs. A pair of these cameras can be used to model a 3D image that provides the information necessary to calculate complex depth information of targets. While mono and stereo cameras are highly sensitive to visibility conditions, thermal imaging cameras have a range of up to 300 meters and are unaffected by fog, dust or extreme lighting conditions [20].

1.2.1.2 STATE-OF-THE-ART SENSOR DEPLOYMENTS FOR ADAS IN VEHICLES

Table 2 below shows the sensor deployment in leading vehicle manufacturers and OEMs along with their targeted level of autonomy. Ford has tested self-driving cars in adverse weather conditions such as snowy weather and poor lighting conditions. With their target of achieving Level 4 Autonomy, they design autonomous vehicles within pre-mapped, “geofenced” areas. Geofencing allows ADAS to attain higher degree of autonomy by using pre-mapped and highly detailed information from selected routes. While they consider wide range of testing conditions in development, a level 4 autonomy model is currently not available for sale in the market in 2021 [21].

Table 2: Leading OEM sensor deployment for autonomy goals

OEM	Sensor selection	Sensor vendors	Target level of autonomy	Test fleet
Ford	Lidar, GPS	Argo, Velodyne	Level 4	Fusion Hybrid
Renault-Nissan	Radar, Lidar	Transdev	Level 3	Commercial cars
VW Audi group	Lidar, Camera	Delphi, Aurora	Level 4	Commercial cars
BMW	Lidar, Camera, GPS	Baidu, Intel	Level 5	Commercial cars
Waymo	Lidar, Radar, Camera	Velodyne, Fiat-Chrsyler	Level 5	Pacifica minivan
Tesla	Camera, Radar	Apple, Mobileye, NVIDIA	Level 5	Commercial cars
Hyundai	Lidar, Camera	KIA, Aurora	Level 3	Commercial cars

Nissan deployed its first Level 2 ADAS called ProPilot on the Serena van model which was made available to the Japanese market in 2016. This initial version of ProPilot offered lane keeping and cruise control abilities only for single lane driving in Japan. With addition of the Intel Mobileye camera to their sensor configuration, the newer version of ProPilot built in 2017 has the ability to change lanes automatically in urban areas. Currently, after collaboration with Renault they are targeting more advanced features in level 3 autonomy [22].

The first self-driving prototype built by Volkswagen-Audi group was the V-charge golf cart in 2015 which used sensors and 3D mapping for localization and navigation within parking spaces in a garage. In 2016, they shifted their development focus to ADAS for mobility services like fleet based on demand transportation. They successfully integrated level 2 autonomy features for the Audi A6, A7 and A8 models. These models were made available to the market in 2020 with a future target of level 3 autonomy by 2025 for all their vehicle models [23].

In 2016, BMW begun their alliance with Intel acquired Mobileye to change its strategy for achieving autonomy. The sensor configuration used involves usage of 2 laser scanners surfaced engineered for the front and rear bumpers respectively. The front camera used is placed at the top of the windshield, while there are separate cameras assigned for rear window road marking detection and road sign detection respectively. Each side mirror with 180 degree FOV uses laser scanners that monitor the left and right spaces around the ego vehicle. Four ultrasonic sensors above the wheel monitors are used for perception while using the park assist feature [24].

Waymo (a subsidiary of Alphabet Inc., originally started as a project by Google in 2009) combines 3 different types of lidar sensors, 5 radar sensors, and 8 cameras [25]. Tesla's vehicles avoid lidars due to their high costs and instead their Autopilot uses 8 surround cameras, 12 ultrasonic sensors (primarily for short-range self-parking support) and 1 forward-facing radar. Each of the cameras has a maximum visibility range of up to 250 meters, so this configuration ensures a 360-degree coverage up to 250 meters around the vehicle [26]. Waymo has stopped using off-the-shelf lidar sensor hardware and begun custom design and manufacture all types of sensors. Waymo's presence in the market will be of an OEM to the auto industry. Waymo intends to market its automated driving stack and provide mobility services to consumers, while Tesla targets to sell vehicle equipped with level 4 autonomy by 2022 [25].

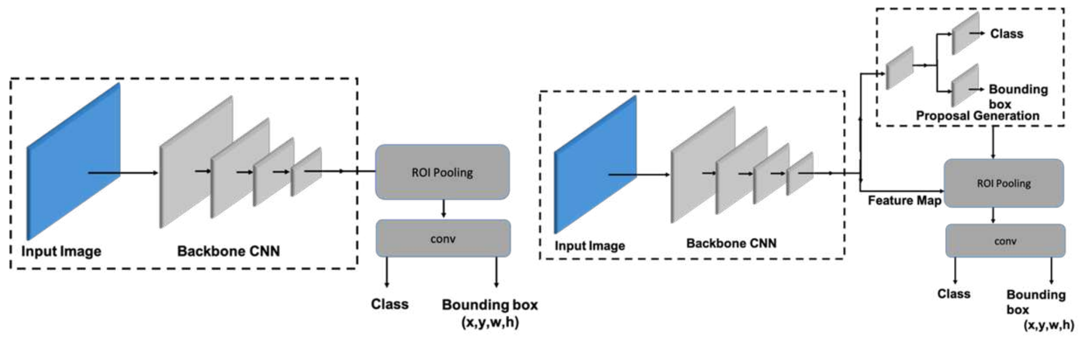
The latest edition of Hyundai's ADAS development module is largely based on the newly launched Ioniq battery EV and is configured with a unique sensor array composed of three front lidar sensors, both short and long-range radars, and a variety of cameras for vision. Effective pedestrian detection, lane tracking, and reading traffic signals is made possible from effective fusion of sensor data and HD maps. The lane keeping system in the 2016 Hyundai Elantra compact is remarkably capable of efficiently detecting lane markings at speeds above 35 mph and applying autonomous steering control to center the car within lane lines [27].

1.2.2 OBJECT DETECTION IN ENVIRONMENT PERCEPTION FOR ADAS

Object detection provides a semantic understanding of the environment from acquired raw camera data. In order to develop a reliable autonomous vehicle capable of perceiving the surrounding environment, selecting an accurate object detector is essential. The diverse driving scenarios impose different object detection challenges due to weather conditions, lighting conditions, extreme elevations or road geometries. In autonomous driving, objects can be classified into static and dynamic objects. Static objects include traffic lights and signs while dynamic objects include pedestrians, cyclists, and different non ego vehicles. Detection of static objects is considered a straightforward task because of their definite shape and easily predicted location, however dynamic object detection is more difficult and allows for a lower margin for detection error on account of their unpredictable velocities or vicinity to the ego vehicle.

Due to the diversity of object classes, lighting, and background conditions, manual feature extraction is not a robust approach. Therefore, deep learning based object detectors play a significant role in object detection tasks as they can learn and extract more complex features and eliminate the need for manually designing features through training of deep neural networks (DNN). There are 2 primary goals associated with deep learning based object detectors for a given image. Initially, the spatial information (relative position of the object in the image) of the target is calculated, which is referred to as *localization,* followed by identifying which category that object instance belongs to, which is referred to as *classification* [28]. The most widely used metric to evaluate the detection accuracy of object detectors is the Average Precision (AP). Further, the Intersection-over-Union (IoU) metric evaluates localization performance since it is a measure of how much the bounding box of the detector's prediction and the ground truth overlaps [28]. A predicted object is considered true positive if the evaluated IoU is above a specific threshold value while it is considered false positive if it is below the value.

The pipeline of traditional object detection models can be mainly divided into informative region selection, feature extraction, and classification. Depending on which subset of these steps are used to process an input image frame, object detectors can be classified as one stage and two stage detectors [29].



(a) (b)

Figure 2: (a): One stage detector architecture; (b): Two stage detector architectures

One-stage detectors, as shown in figure 2(a) are usually composed of a single feed-forward fully convolutional network (CNN) that directly places the anchor/bounding boxes and perform object classification. The backbone CNN is a feedforward neural network designed for feature extraction. Features extracted by the backbone CNN from different spatial regions of the image where objects may be present, are pooled together to identify their class and determine their position using these bounding boxes [29]. The bounding box is described using 4 parameters: (x, y, w, h) where (x, y) denotes the position of the geometric center of the constructed box while w and h describe the width and height respectively.

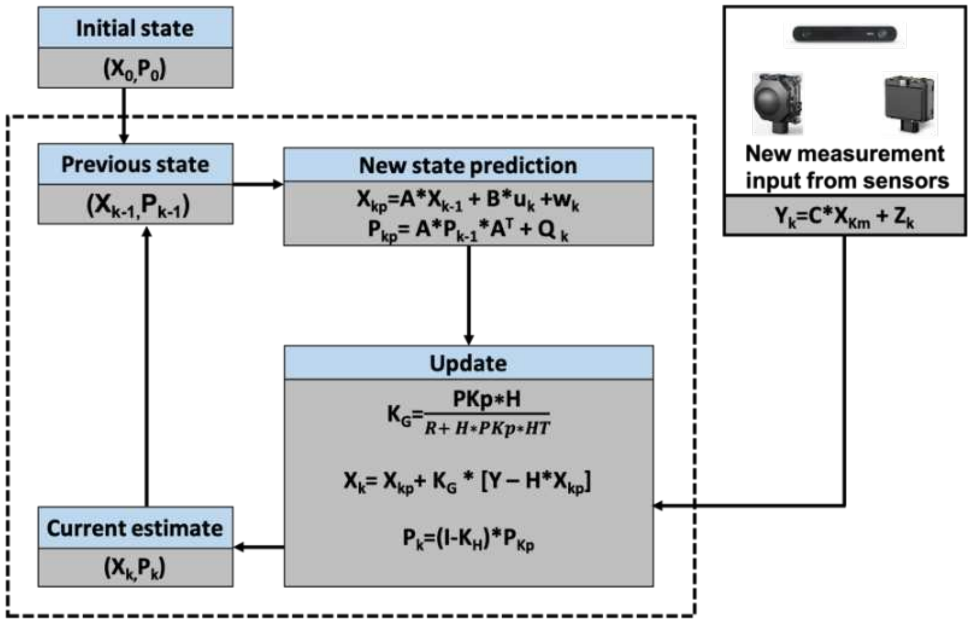
Two-stage detectors shown in figure 2(b) divide the detection process into the region proposal and the classification stage. These models first propose several object candidates, known as Regions of Interest (RoI), using reference boxes (anchors) followed by the second step where the proposals are classified and their localization is refined [29].

The number, position and dimensions of the selected regions in an image have a direct effect on the efficiency of localization. Exhaustively generating regions for the entire image is computationally expensive and may increase inference latencies, while insufficient number of region proposal can cause the object detector to miss objects. State-of-the-art object detectors often perform feature extraction and classification using Convolutional Neural Networks (CNN). However, CNNs alone cannot detect the number of category instances within an image to make accurate detections in real time scenarios where the number of target objects like vehicles in traffic and obstacles are highly dynamic, warranting the need for deep learning based object detectors that combine CNNs with other components for object localization and detection.

1.2.3 SENSOR FUSION FOR TARGET TRACKING IN ADAS

Perception architectures that use multiple sensors in their heterogeneous sensor set often introduce error due to imprecise measurement. Conversely, errors can also arise when only a single sensor is used due to measurement uncertainties from insufficient spatial *(occlusion)* or temporal (*delayed sensor response time*) coverage of the environment. This motivates the need for frameworks that can process data from multiple sensors with minimum error.

Multi Sensor Data Fusion (MSDF) frameworks can be classified into three primary approaches depending on how sensory data is combined from various sensing modalities: high-level fusion (HLF), low-level fusion (LLF) and mid-level fusion (MLF) [30]. In the HLF approach, each sensor first carries out object detection independently and fusion is subsequently performed on the cluster of detections made. High level fusion is often adopted due to a relatively lower computational complexity than the LLF and MLF approach [30]. However, HLF is susceptible to generating inadequate information, since classifications with a lower confidence value are discarded if there are several overlapping obstacles. Conversely, with the LLF approach, raw data from each sensor is fused at the lowest level of abstraction, so there is no loss of information which can result in higher accuracy for object detection. MLF is an abstraction level between LLF and HLF and is commonly referred to as feature level fusion. It fuses important multi-target features extracted from the corresponding raw sensor data such as positional data from radar and color information from RGB frames collected by the camera [31].



Initial state
(X_0, P_0)
Previous state
(X_{k-1}, P_{k-1})
New state prediction
$X_{kp} = A*X_{k-1} + B*u_k + w_k$
$P_{kp} = A*P_{k-1}*A^T + Q_k$
New measurement input from sensors
$Y_k = C*X_{Km} + Z_k$
Update
$K_G = \frac{PKp*H}{R+H*PKp*HT}$
$X_k = X_{kp} + K_G * [Y - H*X_{kp}]$
$P_k = (I-K_H)*P_{Kp}$
Current estimate
(X_k, P_k)

Figure 3: Working of Kalman filter family

Sensor fusion can also be classified according to the algorithm used for state estimation/tracking. The Kalman filter proposed by Rudolf E. Kalman in 1960 is the most widely used state estimation algorithm in autonomous driving and robotics applications for tracking targets [32]. The Kalman filter family are a set of recursive mathematical equations that provide an efficient computational solution of the least-squares method for estimation. They support estimations of past, present and future states, even when the precise nature of the modeled system is unknown [32].

The figure 3 above shows the general working of filters in the Kalman family and explains the important parameters and computations involved in the new state prediction and update steps. The parameters critical to functioning of the Kalman filter listed in the above figure 3 are: the state covariance matrix (P) which is a measure of the error in estimation, the measurement covariance matrix (R) which is a measure of error in measurement, the process noise covariance matrix (Q) inherent to the model of the system and the K_G (Kalman gain) which is a weight factor that compares which error to trust more between: *error in the estimate* versus *error in* *measurement*. Due to the fast convergence property of Kalman filters, the initial state (X_0, P_0) can be selected at random and does not affect the final estimate. A, B and u used in the prediction step consists of velocity and accelerations constants, which are derived from Newton's equation of motion. These are used to compute the new state (X_{kp}) as well as the new error in estimation (P_{kp}) of the state covariance matrix. Next, in the update step the Kalman gain K_G is computed as the fraction of error in estimate over the total error (total error = sum of error in measurement and error in estimate). The value of K_G lies in the range 0 to 1 and is the deciding factor for which of the errors to trust more in computation of updated state X_k. If the computed value of K_G is close to 0 then the estimates are stable and measurements are inaccurate, where as if K_G is closer to 1 then the measurements are accurate and the estimations are unstable. Since, the term $[Y - H*X_{kp}]$ represents the deviation of prediction at a previous time step from the new sensor reading at the current time step, K_G determines what fraction of this difference is added as a correctional factor to update X_{kp} to X_k in the update step.

The Kalman filter has the ability to obtain optimal statistical estimations when the system state is described as a linear model and the error can be modeled as Gaussian noise [32]. If the system state is represented as a nonlinear dynamic model as opposed to a linear model, a modified version of the Kalman filter known as the Extended Kalman Filter (EKF) can be used, which provides an optimal approach for implementing nonlinear recursive filters. However, the EKF has disadvantages due to the computation of the Jacobian (matrix describing the state of the system) being computationally expensive [33]. Further, any attempts to reduce the cost through techniques like linearization make the performance unstable. The Unscented Kalman filter (UKF) has gained popularity, due to its ability to be implemented in parallel, as well as the absence of linearization step and associated errors of the EKF. In contrast to the linearization strategy used by the EKF, the UKF uses a sampling strategy to establish the minimum set of points around the mean referred to as sigma points. These points are propagated through nonlinear functions and the covariance of the estimations can be regained [33].

1.2.3.1 SENSOR FUSION CLASSIFICATION

Depending on the relative position and orientation of the fixed sensor configuration in a perception architecture, sensor fusion can be classified into complementary fusion, redundant fusion and cooperative fusion [34]. As shown below in figure 4, the ego vehicle depicted in blue refers to the vehicle on which perception is enabled by placement of sensors in different configurations. Complementary fusion involves positioning of sensors such that there is no overlap in their field of view resulting in a maximization of coverage of the environment.

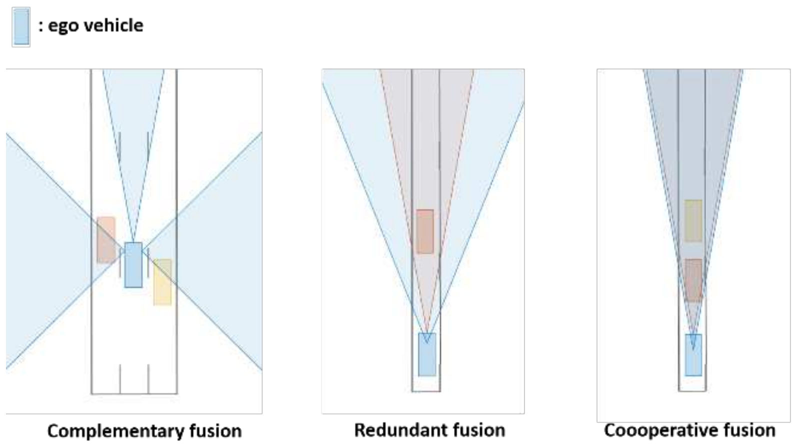


Figure 4: Whyte's classification

Redundant fusion, also referred to as competitive fusion, position sensors such that they focus on the same field of view ensuring higher reliability and detection accuracy. Cooperative fusion involves sharing a common field of view between 2 sensor modalities such that a new representation of the environment can be created to perform detection or depth estimation more efficiently [34].

1.3 CHALLENGES IN SAE LEVELS OF AUTONOMY FOR ADAS

1.3.1 SAE LEVELS OF AUTONOMY

The degree to which ADAS can effectively reduce human intervention during driving is classified by Society of Automotive Engineers (SAE) according to the J3016 standard, into six levels of autonomy as shown in figure 5 below [35].

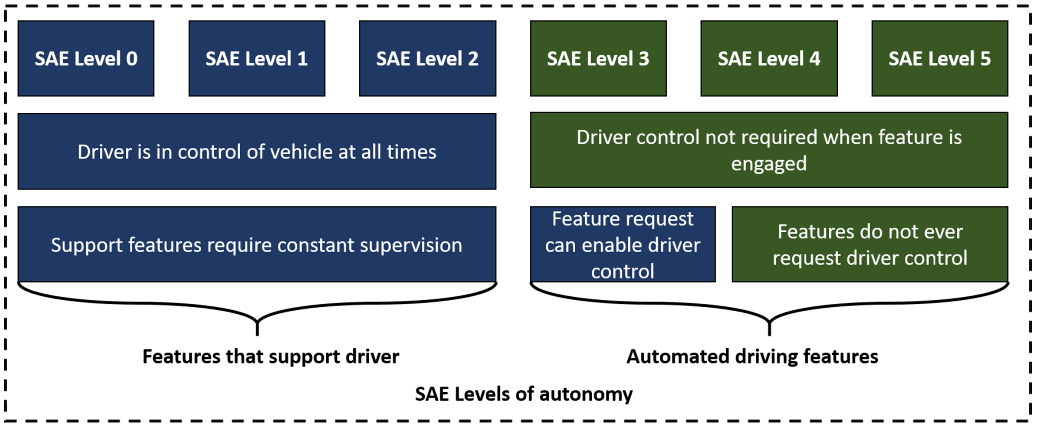

SAE Level 0
SAE Level 1
SAE Level 2
SAE Level 3
SAE Level 4
SAE Level 5
Driver is in control of vehicle at all times
Driver control not required when feature is engaged
Support features require constant supervision
Feature request can enable driver control
Features do not ever request driver control
Features that support driver
Automated driving features
SAE Levels of autonomy

Figure 5: SAE levels of autonomy

Level 0 characterizes vehicles that have no assistive features. Level 1 autonomy encompasses vehicles that have the ability to share control between the driver and the vehicle. Adaptive cruise control and park assist are examples of features that can assist the driver in this level [35]. Level 2 autonomy vehicles have the capability to perform all acceleration, steering and braking tasks that require longitudinal and lateral control. Examples of features supported in this level include forward collision warning and blind spot warning, in addition to features from level 1. Level 3 autonomy vehicles can assess the risk of a situation and additionally perform path planning. At Level 4 autonomy, no driver intervention is required in most cases, unless requested, in contrast to level 3. Level 5 autonomy requires no human intervention or safety driver in the vehicle, unlike in level 4 [35].

1.3.2 CHALLENGES IN LEVEL 2 AUTONOMY FEATURES

As our research focuses on supporting level 2 perception autonomy, here we discuss the requirements at this level. A perception architecture designed to support Level 2 autonomy in a vehicle should support all four of the critical features discussed in the following sections.

1.3.2.1 ADAPTIVE CRUISE CONTROL (ACC)

In 1992, Mitsubishi was the first to offer a distance detection system called Debonair which used a lidar for "distance warning" without influencing any control functions [36]. Later in 2000, Toyota introduced their *adaptive cruise control (ACC)* system to the US market which used a LS 430 Dynamic Laser Cruise Control system and did not require any input from the driver once the feature was engaged [37]. ACC causes the ego vehicle to follow a lead vehicle at a specified distance without exceeding the speed limit specified by the operator upon activation of the feature [38]. If the lead vehicle slows down, then it is the responsibility of ACC to slow down the ego vehicle to maintain the specified distance. Although implementations differ, all ACC systems take over longitudinal control from the driver. The challenge in ACC is to maintain an accurate track of the lead vehicle with a forward facing sensor and using longitudinal control to maintain the specified distance while maintaining driver comfort (e.g., avoiding sudden velocity changes).

1.3.2.2 LANE KEEP ASSIST (LKA)

State-of-the-art *lane keep assist (LKA)* feature variants are an evolution of lane departure warning systems. Data from a forward facing camera is often processed using Canny edge detection or Hough transform to identify lane line information in front of the ego vehicle [39]. Using this information LKA can determine whether the ego vehicle is drifting towards any of the lane boundaries and respond by sending steering torque commands to the control system until it has no trajectory to cross. However, LKA systems have been known to over-compensate, creating a “ping-pong” effect where the vehicle oscillates back and forth between the lane lines [40]. The main challenges in LKA are to reduce this ping-pong effect and the accurate detection of lane lines on obscured (e.g., dirt covered) roads.

1.3.2.3 FORWARD COLLISION WARNING (FCW)

Forward collision warning (FCW) uses information gathered via various forward facing sensors for real time prediction of collisions with a lead vehicle. The driver is notified to apply brakes through an audio visual warning when the ego-vehicle is dangerously close to the lead vehicle. As this is a safety-critical system, it is important that FCW avoids false positives as well as false negatives to improve driver comfort, safety, and reduce rear end accidents [41]. For this to be achieved, it is a necessary prerequisite that the sensors used by the FCW system be placed where they have an accurate view of the vehicle in front of them. The United States National Transportation Safety Board has recommended that FCW be included in all new vehicles [42].

1.3.2.4 BLINDSPOT WARNING (BW)

Blind spot warning (BW) uses sensors mounted on the lateral sides of the ego vehicle to determine whether there is a vehicle towards the rear on either side of the ego vehicle in a location the driver cannot see with their side mirrors [43]. This area is commonly referred to as the “blind spot”, requiring the driver to turn their head and verify whether there are oncoming vehicle during a lane change. With BW, the driver can maintain their concentration on the road ahead and perform lane changes without having to look behind. As BW requires information about a specific area near the rear of the vehicle, it is a challenge to find an optimal sensor placement that maximizes the view of the blind spot. If the sensor is too far forward, it will miss the blind spots entirely, causing a vehicle accident when the driver makes a lane change. If the sensor is too far back, it will end up capturing information for areas around the ego vehicle that are not in the blind spot, decreasing the sensor’s effectiveness at viewing the presence of vehicles surrounding the blind spot.

1.4 DISSERTATION OVERVIEW

In summary, there is a crucial need for a holistic framework in perception architecture synthesis for ADAS. Such a framework is not trivial to conceptualize, because of the complex inter-dependencies between design decisions, constraints, and optimization goals. For example, to enhance obstacle detection for a given feature there is an inherent trade-off between accuracy and latency for one stage and two stage deep learning based detectors. Further, even if a perception architecture is equipped with an ideal object detector having high accuracy and low latency, performance is significantly affected by the position and orientation of the sensor determining the combined field of view. If the combined field of view is too small or contains redundant overlap between individual sensors, important events and obstacles can go undetected while if the field of view is too large, the number of false positive detections will be high and appropriate sensor fusion algorithms are needed for target tracking to reduce the number of false positive detections.

To address the above-mentioned problems, the main contribution of this dissertation is the design of a novel exploration framework for perception architecture exploration in automotive cyber-physical platforms. This framework aims to enhance the performance of perception in

ADAS, while meeting a diverse set of real time driving constraints (e.g. reliability, latency, fidelity, weather resiliency, and adaptability to road geometries).

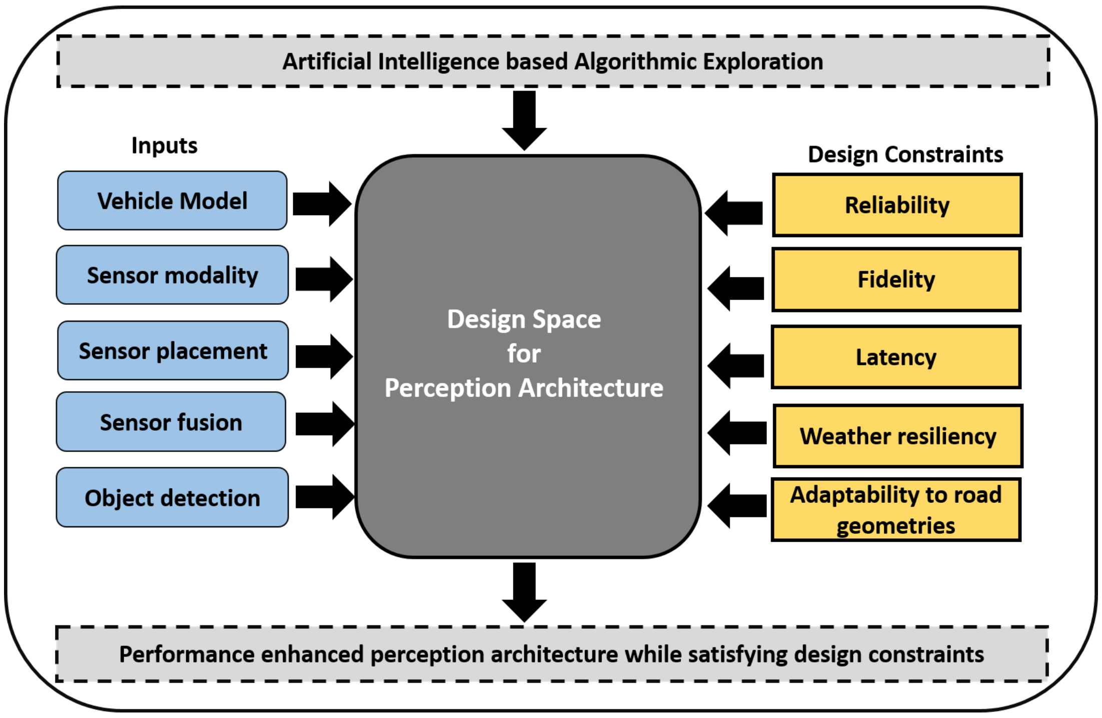

Artificial Intelligence based Algorithmic Exploration
Inputs
Vehicle Model
Sensor modality
Sensor placement
Sensor fusion
Object detection
Design Space for Perception Architecture
Design Constraints
Reliability
Fidelity
Latency
Weather resiliency
Adaptability to road geometries
Performance enhanced perception architecture while satisfying design constraints

Figure 6: Overview of perception architecture exploration framework for automotive cyber-physical platforms

Figure 6 shows a high level overview of the proposed framework, with our main contributions describing various facets of this framework explained in detail in chapters 2 and 3. This framework performs intelligent algorithmic exploration in the design space of perception architecture to synthesize reliable perception architectures that are capable of high fidelity environment modeling and information extraction. The framework is sensitive to the selected input vehicle model. The design space of possible perception architecture solutions are the total number of ways in which the following inputs can be selected for a given vehicle model: sensor modality selection, sensor placement, object detectors, and sensor fusion algorithm. This design space is vast due to combinatorial explosion and cannot be explored exhaustively in a practical period of time to yield solutions, warranting the need for intelligent exploration. The use of artificial intelligence based search algorithms and evolutionary algorithms enables efficient exploration in our framework while satisfying design constraints ensuring latency, reliability, fidelity and weather resiliency. The rest of this dissertation is organized as follows:

In chapter 2, we address the synthesis of heterogeneous sensor configurations towards achieving vehicle autonomy goals. We propose a novel optimization framework called VESPA that explores the design space of sensor placement locations and orientations to find the optimal sensor configuration for a vehicle. We demonstrate how our framework can obtain optimal sensor configurations for heterogeneous sensors deployed across two contemporary real vehicles.

In chapter 3, we address the multiple complex problems related to the selection and placement of sensors, design of object detector, and sensor fusion. Current methods address these problems in isolation, which leads to inefficient solutions. We present PASTA, a novel framework for global co-optimization of deep learning and sensing for ADAS-based vehicle perception. Experimental results with the Audi-TT and BMW Minicooper vehicles show how PASTA can intelligently traverse the perception design space to find robust, vehicle-specific solutions.

Chapter 4 concludes this dissertation. We summarize our comprehensive body of research in this chapter and also make recommendations for future work.

2. VESPA: A FRAMEWORK FOR OPTIMIZING HETEROGENEOUS SENSOR PLACEMENT AND ORIENTATION FOR AUTONOMOUS VEHICLES

In emerging autonomous vehicles, perception of the environment around the vehicle depends not only on the quality and choice of sensor type, but more importantly also on the instrumented location and orientation of each of the sensors. This chapter explores the synthesis of heterogeneous sensor configurations towards achieving vehicle autonomy goals. We propose a novel optimization framework called VESPA that explores the design space of sensor placement locations and orientations to find the optimal sensor configuration for a vehicle. We demonstrate how our framework can obtain optimal sensor configurations for heterogeneous sensors deployed across two contemporary real vehicles.

2.1 MOTIVATION AND CONTRIBUTION

The increasing maturity of Advanced Driver Assistance Systems (ADAS) is enabling the introduction of vehicles with greater levels of autonomy. The degree to which ADAS can effectively reduce human intervention during driving is classified by SAE according to the J3016 standard [44], into five levels of autonomy.

Level 0 characterizes vehicles that have no assistive features. Level 1 autonomy encompasses vehicles that have the ability to share control between the driver and the vehicle. Adaptive cruise control and park assist are examples of features that can assist the driver in this level. Level 2 autonomy vehicles have the capability to perform all acceleration, steering, and braking tasks that require longitudinal and lateral control. Examples of features supported in this level include forward collision warning and blind spot warning, in addition to features from level 1. Level 3 autonomy vehicles can assess the risk of a situation and additionally perform path planning. At Level 4 autonomy, no driver intervention is required in most cases, unless requested, in contrast to level 3. Level 5 autonomy requires no human intervention or safety driver in the vehicle, unlike in level 4. *Most vehicles today are beginning to support level 2 autonomy.*

The higher autonomy levels require support for increasingly sophisticated ADAS features such as Lane Keep Assist (LKA) and Forward Collision Warning (FCW), which in turn defines requirements for sensing capabilities and perception performance of the vehicle. Table 3 summarizes the trade-offs between popular sensors used to support ADAS features and their relative performance. Using a camera as a vision sensor is a widely used approach to perform classification and detection of objects on the road. However, cameras have high susceptibility to noise and are not reliable in extreme weather or lighting conditions [45]. A *radar* sensor is also capable of object detection and is particularly suited for accurate velocity detection of neighboring vehicles even under harsh weather and poor visibility conditions. Long-range radars (typically at 77GHz) used to support ADAS features such as adaptive cruise control (ACC) and automatic emergency braking (AEB) have a shorter azimuth than mid or short-range radars (typically at 24GHz), to prioritize monitoring vehicle velocity and approaching distance. However, long range radars can also detect more number of objects than short or mid-range radars. A drawback of the radar is their high false positive rate when detecting objects, and an upper bound on the number of objects that can be detected at the same time, e.g., the Bosch midrange radar with a maximum range of 160 meters can only detect up to 32 objects simultaneously [46]. A *LiDAR* sensor uses invisible laser light to measure the distance to objects in a similar way to radars. It can create an incredibly detailed 3D view (point cloud) of the environment around the vehicle. However, LiDAR data processing is computationally very expensive and relies on moving parts which can make it more vulnerable to damage. Ultrasonic sensors listed in table 3 use the principle of 'time of flight' to measure distance from targets by computing the travel time of the ultrasonic echo from a neighboring vehicle or obstacle [47]. Usage of ultrasonic sensors for ADAS feature implementation are not uncommon, however they require accurate modelling for their use case, since their performance is highly dependent on the physical properties (shape, surface material) of the target being tracked [48].

Table 3: ADAS sensor trade-offs

Characteristics	Camera	LiDAR	Radar
Perception Reliability	Medium	High	Medium
Spatial Resolution	High	High	Low
Noise Susceptibility	High	Low	Low
Velocity Detection	Low	Low	High
Weather Durability	Low	Low	High

Most level 2 and higher autonomy vehicles today rely on a combination of sensors, to overcome their individual drawbacks (see Table 3). For example, Waymo (a subsidiary of Alphabet Inc., originally started as a project by Google in 2009) combines 3 different types of LiDAR sensors, 5 radar sensors, and 8 cameras. Tesla's vehicles avoid LiDARs due to their high costs and instead their Autopilot uses 8 surround cameras, 12 ultrasonic sensors (primarily for short-range self-parking support), and 1 forward-facing radar. Each of the cameras has a maximum visibility range of up to 250 meters, so this configuration ensures a 360-degree coverage up to 250 meters around the vehicle.

An important challenge facing emerging vehicles is to determine a sensor configuration that can be responsible for environment perception as per the SAE autonomy level supported by the vehicle. An optimal sensor configuration should consist of carefully selected *location* and *orientation* of each sensor in a heterogeneous suite of sensors, to maximize coverage from the combined field of view obtained from the sensors, and also maintain a high object detection rate. Today there are no generalized rules for the synthesis of sensor configurations, as the location and orientation of sensors depends heavily on the target features and use cases to be supported in the vehicle.

In this chapter, we propose a novel framework called VESPA (VEhicle Sensor Placement and orientation for Autonomy), to optimize heterogeneous sensor synthesis. More precisely, for a given set of heterogeneous sensors and ADAS features to be supported, VESPA performs intelligent algorithmic design space exploration to determine the optimal placement and orientation for each sensor on the vehicle, to support the required ADAS features for SAE level 2 autonomy systems. The VESPA framework can be easily utilized to generate optimal sensor configurations across different vehicle types. Our experimental results indicate that the proposed framework is able to optimize perception performance across multiple ADAS features for the 2019 Chevrolet Blazer and 2016 Chevrolet Camaro vehicles.

2.2 RELATED WORK

State-of-the-art SAE level 2 autonomy systems require the selection and placement of sensors based on the assistive target features required to be supported, e.g., forward collision warning (FCW) and lane keep assist (LKA). While several prior works evaluate the performance of a specific sensor configuration and its deployment, very few works have explored the problem of generating optimal sensor configurations for vehicles.

An optimal sensor placement approach was proposed in [49] for a blind spot detection and warning system. The work recognizes the inability of the camera to perform in non-ideal lighting conditions and selects an ultrasonic sensor to measure distance of vehicles trailing in the vehicle's blind spot. The time response of the system with the position of the sensor above the rear tire is analyzed for two scenarios: when the vehicle is at rest and when it is moving at a constant velocity. The sensor selection identifies price as a constraint and optimizes the price of the total sensor setup through usage of an ultrasonic sensor instead of a more expensive camera sensor. The work in [50] focuses on generating a LiDAR configuration from a set of LiDARs with the goal of reducing occurrences of dead zones and improving point cloud resolution. A LiDAR occupancy grid is constructed for a homogenous set of LiDARs and the configuration is generated using a genetic algorithm. An approach for optimal positioning and calibration of a three LiDAR system is proposed in [51] that uses a neural network to qualify the effectiveness of different sensor location and orientations. Unlike these prior works that focus on generating configurations for a homogenous set of sensors, our work in this chapter presents a novel sensor placement and orientation optimization framework for a heterogeneous set of sensors. Moreover, our framework is also shown to be capable of easily adapting to different vehicle types.

2.3. BACKGROUND

We target four ADAS features in this chapter that need to be supported by a deployed sensor configuration on a vehicle (henceforth referred to as an *ego vehicle*). A sensor configuration consists of the location and orientation of each sensor within a heterogeneous set of sensors. Our VESPA framework optimizes the sensor configuration to support four features: adaptive cruise control (ACC), lane keep assist (LKA) forward collision warning (FCW), and blind spot warning (BW). Each of the features discussed above, require varying degrees of sensing and control along longitudinal (i.e., within the same lane as the ego vehicle) and lateral (i.e., along neighboring lanes) regions.

2.3.1 ADAS FEATURES FOR LEVEL 2 AUTONOMY

SAE J3016 defines ACC and LKA individually as level 1 features, as they only perform the dynamic driving task in either the latitudinal or longitudinal direction of the vehicle. FCW and BW are defined in SAE J3016 as level 0 active safety systems, as they only enhance the performance of the driver without performing any portion of the dynamic driving task. However, when all four features are combined, the system can be described as a level 2 autonomy system. Many new vehicles being released today support level 2 autonomy. For instance, Volvo announced that its upcoming Level 2+ vehicles will use surround sensors for 360-degree perception, as well as deep neural networks running in parallel for robust object detection [52]. It is not only relevant, but also important to optimize sensor placement for ADAS systems as more and more vehicles with these features become available. Figure 7 shows an overview of the four features we focus on for level 2 autonomy, which are discussed next.

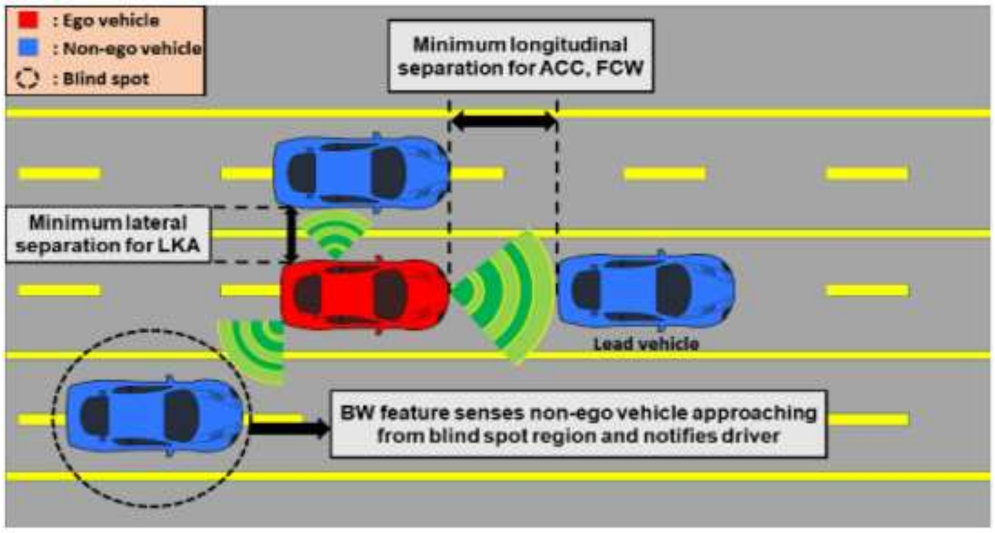

🟥 : Ego vehicle
🟦 : Non-ego vehicle
◌ : Blind spot
Minimum longitudinal separation for ACC, FCW
Minimum lateral separation for LKA
Lead vehicle
BW feature senses non-ego vehicle approaching from blind spot region and notifies driver

Figure 7: Visualization of common scenarios in ACC, FCW, LKA, and BW

Adaptive cruise control (ACC) was first introduced in the Mercedes-Benz S-Class sedan in 1999, with the goal of increased driver comfort. ACC causes the ego vehicle to follow a lead vehicle at a specified distance (figure 7) without exceeding the speed limit specified by the operator upon activation of the feature [53]. If the lead vehicle slows down, then it is the responsibility of ACC to slow down the ego vehicle to maintain the specified distance. Although implementations differ, all ACC systems take over longitudinal control from the driver (figure 7). The challenge in ACC is to maintain an accurate track of the lead vehicle with a forward facing sensor and using longitudinal control to maintain the specified distance while maintaining driver comfort (e.g., avoiding sudden velocity changes).

Lane keep assist (LKA) is an evolution of lane departure warning systems. It involves a forward-facing sensor (often a camera) to identify where the lane lines exist in front of the ego vehicle. Once the lane lines have been detected (e.g., using Canny edge detection and Hough transforms on forward-facing images), LKA can then determine if the ego vehicle lies between those lines (figure 7). If the ego vehicle appears to be drifting toward a position where it will cross lane line boundaries, LKA engages steering torque to steer the vehicle in the opposite direction of the lane line until it no longer has the trajectory to cross that lane. LKA systems have been known to over-compensate, creating a "ping-pong" effect where the vehicle oscillates back and forth between the lane lines [54]. The main challenges in LKA are to reduce this ping-pong effect and the accurate detection of lane lines on obscured (e.g., dirt covered) roads.

Forward collision warning (FCW) uses information gathered via various forward facing sensors to determine whether the ego vehicle is going to collide with an object in front of it (figure 7). As objects approach the boundary where the vehicle can no longer come to a stop, an audio-visual warning notifies drivers instructing them to apply the brakes. As this is a safety-critical system, it is important that FCW avoids false positives as well as false negatives to improve driver comfort, safety, and reduce rear end accidents [55]. For this to be achieved, it is a necessary prerequisite that the sensors used by the FCW system be placed where they have an accurate view of the vehicle in front of them. The United States National Transportation Safety Board has recommended that FCW be included in all new vehicles [56].

Lastly, blind spot warning (BW) uses sensors mounted on the sides of the ego vehicle to determine whether there is a vehicle towards the rear on either side of the ego vehicle in a location the driver cannot see with their side mirrors [57] (figure 7). This area is typically referred to as the "blind spot" and must be verified as clear of any vehicles before the driver can attempt to make a lane change. Without BW, the driver must turn their head to make that verification on their own. With BW, the driver can maintain their concentration on the road ahead. As BW requires information about a specific area near the rear of the vehicle, it is a challenge to find an optimal sensor placement that maximizes the view of the blind spot. If the sensor is too far forward, it will miss the blind spots entirely, causing a vehicle accident when the driver makes a lane change. If the sensor is too far back, it will end up capturing information for areas around the ego vehicle that are not in the blind spot, decreasing the sensor's effectiveness at viewing the presence of vehicles surrounding the blind spot.

2.3.2 FEATURE PERFORMANCE METRICS

In equations (1)-(8), the *ground truth* refers to the actual position of the non-ego vehicles (traffic in the environment of the ego vehicle).

$$\text{Longitudinal Position Error (m1)} = \frac{\sum(y - ygroundtruth)}{Number\ of\ non\ ego\ vehicle} \quad (1)$$

$$\text{Lateral Position Error (m2)} = \frac{\sum(x - xgroundtruth)}{Number\ of\ non\ ego\ vehicle} \quad (2)$$

$$\text{Object Occlusion Rate (m3)} = \frac{\text{Number of non ego vehicle undetected}}{Total\ number\ of\ passing\ non\ ego\ vehicles} \quad (3)$$

$$\text{Velocity Uncertainty (m4)} = \frac{\text{Number of invalid detected non ego vehicle velocities}}{Total\ number\ of\ non\ ego\ velocities} \quad (4)$$

$$\text{Rate of late detection (m5)} = \frac{\text{Number of late non ego vehicle detection}}{Total\ number\ of\ non\ ego\ vehicles} \quad (5)$$

$$\text{False positive lane detecion rate (m6)} = \frac{\text{Number of false positive lane detections}}{Total\ number\ of\ lane\ detections} \quad (6)$$

$$\text{False negative lane detecion rate (m7)} = \frac{\text{Number of false negative lane detections}}{Total\ number\ of\ lane\ detections} \quad (7)$$

$$\text{False positive object detecion rate (m8)} = \frac{\text{Number of false positive non ego vehicle detections}}{\text{Total number of non ego vehicle detections}} \quad (8)$$

The longitudinal position error *(m1)* and lateral position error *(m2)* are computed as the deviation of the positional data detected by the sensor configuration from the ground truth of non-ego vehicle positions along the y and x axes respectively. The lateral position error is relevant for LKA, while longitudinal position error is most relevant for ACC and FCW. The object occlusion rate *(m3)* measures the percentage of passing non-ego vehicles that go undetected in the vicinity of the ego vehicle. The minimization of this metric optimizes BW capabilities of a sensor configuration. The velocity uncertainty *(m4)* is the fraction of times that the velocity of a non-ego vehicle is measured incorrectly, which matters for ACC and FCW. The rate of late detection metric *(m5)* is computed as a fraction of the number of 'late' non ego vehicle detections made by the total number of non-ego vehicles, which matters for BW. A detection is classified as late if it is made after the non-ego vehicle crosses the minimum safe longitudinal or lateral distance defined by Intel RSS (Responsibility Sensitive Safety) models on NHTSA for pre-crash scenarios [58]. When a lane marker is detected but there exists no ground truth lane in simulation it is classified as a false positive lane detection, conversely, if a ground truth lane exists in simulation but is not detected, it is classified as a false negative lane detection [59]. Metrics 6 and 7 *(m6 and m7)* characterize the perception system's ability to make a correct case for lane keep assist by taking into account the false positive and false negative lane detection rate. False positive object detection rate *(m8)* measures the fraction of total vehicle detections which were classified as non-ego vehicle detections but did not actually exist in ground truth in the test cases.

2.4 VESPA FRAMEWORK

The following section describes the proposed VESPA framework in detail.

2.4.1 OVERVIEW

Figure 8 shows an overview of our proposed VESPA framework. The physical dimensions of the vehicle model and the number and type of sensors to be considered are inputs to the framework. A design space exploration algorithm is used to generate a sensor configuration which is subsequently evaluated based on a cumulative score from the performance metrics presented in the previous section. We evaluate three design space exploration algorithms: simulated annealing with greedy randomized adaptive search (SA+GRASP), genetic algorithm (GA), and particle swarm optimization (PSO). The process of sensor configuration generation and evaluation continues until an algorithm-specific stopping criteria is met, at which point the best configuration is output. The following subsections describe our framework in more detail.

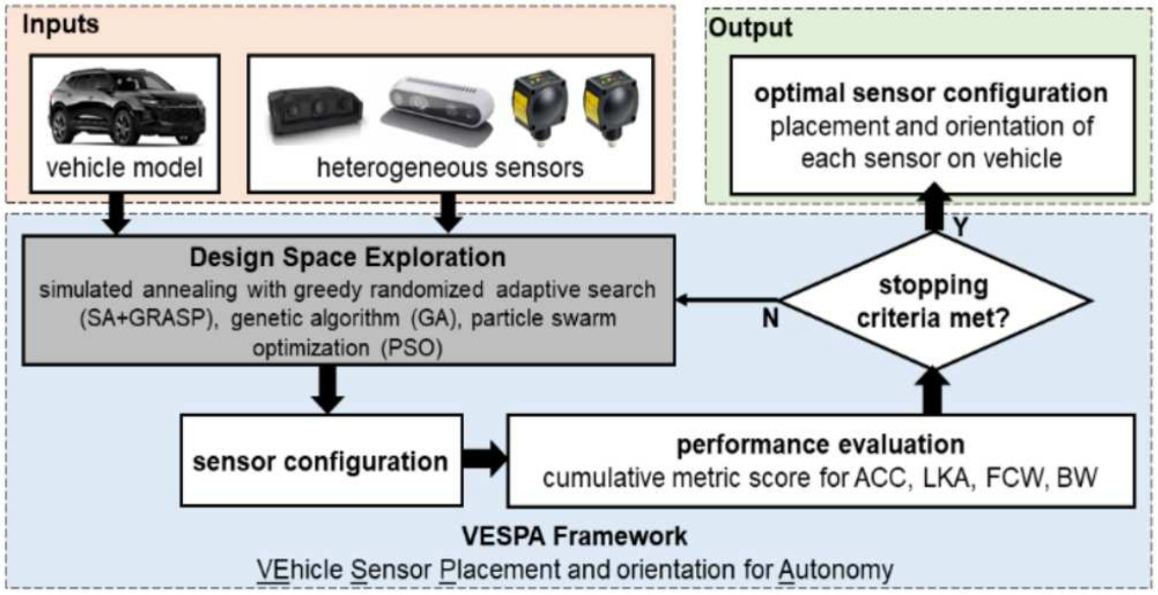
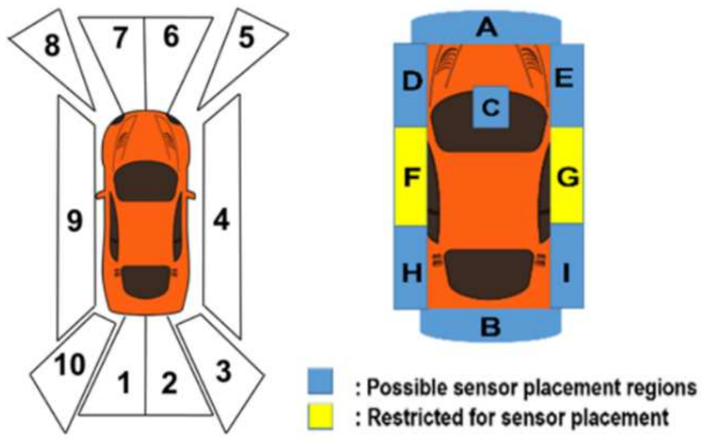

Inputs
vehicle model
heterogeneous sensors
Output
optimal sensor configuration
placement and orientation of each sensor on vehicle
Y
Design Space Exploration
simulated annealing with greedy randomized adaptive search (SA+GRASP), genetic algorithm (GA), particle swarm optimization (PSO)
N
stopping criteria met?
sensor configuration
performance evaluation
cumulative metric score for ACC, LKA, FCW, BW
VESPA Framework
VEhicle Sensor Placement and orientation for Autonomy

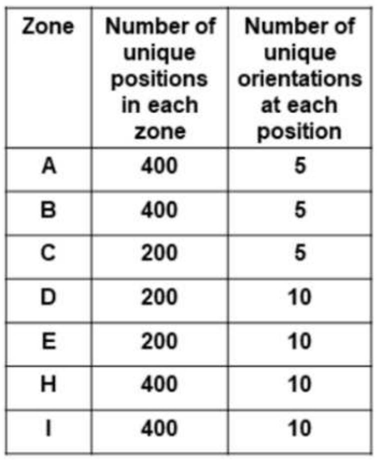
Figure 8: Overview of VESPA framework

2.4.2 INPUTS

Each of the design space exploration algorithms generates sensor configurations that consider feature to field of view (FOV) zone correlations around the ego vehicle. Figure 9(a) shows the FOV zones around the ego-vehicle. These zones of interest are defined as the most important perception areas in the environment for a particular feature. Figure 9(b) shows the regions on the vehicle on which sensors can be mounted (in blue). Regions F and G (in yellow) are exempt from sensor placement due to the mechanical instability of placing sensors on the door of a vehicle.

The correlation between features, zones, regions, and performance metrics shown in Figure 9 is summarized in Table 4. For example, in figure 9(a), for ACC, the zones of interests are 6, and 7, and the corresponding regions for possible sensor placement are A and C. For exploration of possible locations within a region, a fixed step size of 5cm in two dimensions across the surface of the vehicle is considered, which generates a 2D grid of possible positions in each zone shown in figure 9(b), (c). The orientation exploration of each sensor involves rotation at a fixed step size of 1 degree between an upper and lower bounding limit for roll, pitch and yaw respectively, at each of these possible positions within the 2D grid.


8
7
6
5
9
4
10
1
2
3
A
D
C
E
F
G
H
I
B
: Possible sensor placement regions
: Restricted for sensor placement



Zone	Number of unique positions in each zone	Number of unique orientations at each position
A	400	5
B	400	5
C	200	5
D	200	10
E	200	10
H	400	10
I	400	10

(a) (b) (c)

Figure 9: (a) Field of view (FOV) zones; (b) sensor placement regions; (c) design space breakdown

The orientation exploration limits were chosen with caution to the caveat that long range radars with extreme orientations increase the number of recorded false positives. The combined position and orientation exploration generates an intractably large design space as discussed next.

Table 4: Feature, region, zone and performance metric relationship

Feature	Region	Zone	Associated Metrics
BW	B.H.I	1, 2,3,10	(m3, m5, m8)
LKA	E, I	3,4,5	(m2, m3, m6, m7)
	D, H	8, 9, 10	
ACC, FCW	A, B,C	6, 7, 11	(m1, m4, m8)

2.4.3 DESIGN SPACE EXPLORATION

All of the metrics *(m1 – m8)* defined in 2.3.2 represent good performance at lower values. We create a cost function that combines these metrics and frame our sensor placement and optimization problem as a minimization problem. The most important metrics are identified and grouped for each feature, as shown in Table 4, and are used to model the cost function as a weighted sum of these five metrics, where the weights are chosen on the basis of their total cardinality across all feature. By searching through the design space of sensor configurations for a minimum cost function value, a sensor configuration can thus be generated where the metrics are cumulatively minimized.

The design space considered in this chapter uses 4 radars and 4 cameras that can be placed in any zone. With a fixed step size of 5cm in each dimensions and 1 degree rotation in orientation, the number of ways 8 sensors can be placed in all unique locations and orientations is ${}^{2.56e+23}C_8$ for the 2019 Blazer and ${}^{6.4e+22}C_8$ for the 2016 Camaro. As this design space is so large that it cannot be exhaustively traversed in a practical amount of time, we explore the use of intelligent design space search algorithms that support hill climbing to escape local minima. The three algorithms implemented as part of VESPA are discussed next.

2.4.3.1 SA + GREEDY RANDOM ADAPTIVE SEARCH PROCEDURE (SA + GRASP)

Simulated annealing (SA) is a search algorithm that is useful in finding the global optima when the design space has multiple local optima [60]. The process is analogous to the way metals cool and anneal [61]. Typically, SA picks the best solution at each iteration, but can also pick the worst solution based on a temperature-dependent probability, which can allow it to climb out of local minima to arrive at global minima [62]. But SA suffers from the drawback of behaving like a greedy algorithm at lower temperatures as it tends to accepts only those solution configurations very close in cost function value to the previous solution, so it can get stuck in local minima in more complex design spaces [63]. The GRASP (Greedy Randomized Adaptive Search Procedure) algorithm is another search algorithm that is used in many exploration problems [64], but it does not always generate optimal solutions during the greedy construction phase and can get stuck in local optima easily. The SA+GRASP algorithm eliminates the inherent drawbacks of each algorithm. Specifically, the greedy randomized construction phase of the algorithm is used to create disturbances in the existing list of best sensor configurations in our problem, to generate better solutions. A new solution is generated in each iteration by selecting the better solution between the greedy solution from the greedy randomized construction phase and the configuration found from the local search. We decreased the SA temperature variable from T_{max}=10,000 to T_{min}=0 at the rate of 4 degrees per iteration. The search repeats by decreasing SA temperature till an optimal solution is found or a stopping criterion is achieved.

2.4.3.2 GENETIC ALGORITHM (GA)

The GA is an evolutionary algorithm that can solve optimization problems by mimicking the process of natural selection [65]. It repeatedly selects a population of candidate solutions and then improves the solutions by modifying them. GA has the ability to optimize problems where the design space is discontinuous and also if the cost function is non differentiable [66]. The GA is adapted for our design space such that a chromosome is defined by the combined location and orientation of each sensor's configuration (consisting of six parameters: x, y, z, roll, pitch, and yaw). For a given set of N sensors, the number of parameters stored in each chromosomes is thus '6N'. Next, in the selection stage, the cost function values are computed for 100 configurations at a time, and a roulette wheel selection method is used to select which set of chromosomes will be involved in the crossover step based on their cost function probability value, computed as a fraction of the cumulative cost function sum of all chromosomes considered in the selection. In the crossover stage, the crossover parameter is set to 0.5, which allows 50 out of the 100 chromosomes to produce offspring. The mutation parameter is set to 0.2 such that in the mutation stage, the mutation rate is set to 10, which is the number of new genes allowed for mutation in each iteration.

2.4.3.3 PARTICLE SWARM OPTIMIZATION (PSO)

PSO considers a group of particles where each particle has a position and velocity and is a solution to the optimization problem [67]. In our problem each sensor configuration in the design space is represented as a particle having a defined position and velocity. With a random start, the cost function in (5) evaluates the quality of the solution of a particle. The particle's velocity and position values are updated recursively using a linear update [67]. Each particle stores a trace of its best position within the group and globally as well. The history of the cost function values for this trace can explain the effectiveness of changing the position of a particular sensor from the set of heterogeneous sensors [68]. Unlike GA, PSO does not have any evolution operators like crossovers or mutation [69]. PSO also does not require any binary encoding of solution configurations like in GA [70]. The total number of particles considered were 50, and the importance of personal best and importance of neighborhood best parameters were both empirically selected to be 2.

2.5 EXPERIMENTS

The following section describes the experimental setup and results involving the VESPA framework.

2.5.1 EXPERIMENTAL SETUP

To evaluate our VESPA framework, we consider a scenario with a maximum of 8 sensors: 4 radars and 4 camera vision sensors. Many recent contributions such as the work presented in [71] and [72] combine radar and camera modalities for ADAS applications. We did not include LiDARs in this heterogeneous set of sensors due to their relatively poor performance in adverse weather conditions as shown in Table 3. For the given set of test cases, it was observed that if less than 4 sensors were used, the ability of the perception system to make an accurate prediction was relatively poor. Conversely, on increasing the number of radars and cameras to more than 4 each, there was minimal improvement in cost function score. Hence to keep implementation cost low while still achieving good accuracy, we decided to use these 8 sensors. Please note that these modalities and number of sensors have been used to show a proof of concept for our VESPA framework, which can be extended to scenarios with different modalities and numbers of sensors. We considered two vehicles for evaluation: a 2019 Chevrolet Blazer and a 2016 Chevrolet Camaro. Figure 10 shows the dimensions for the vehicles. Figure 11 shows images of the sensor placements on both car models in our workspace.

Each configuration generated by the SA+GRASP, GA, and PSO algorithms was optimized on 40 test cases designed (10 test cases each for evaluating performance with ACC, FCW, LKA, and BW) using the Automated Driving Toolbox in Matlab. Half (20) of these test cases for each feature are used during the optimization phase and the remaining (20) test cases are used during the evaluation phase.

	Length	Width	Height	Front Overhang	Rear Overhang
Blazer	4.7498	1.9558	1.6764	1.016	0.9144
Camaro	4.7498	1.905	1.3716	0.9652	1.016

Figure 10: 2019 Chevrolet Blazer (Left) and 2016 Chevrolet Camaro (Right)

Finally, the optimized configurations were evaluated on a different set of evaluation test cases. Each of the test cases was characterized by unique road geometry, variations in road elevation, curvature, banking, and different traffic densities. In some test cases, the number of lanes were varied to make the framework optimize the sensor configuration for challenging and realistic driving scenarios.

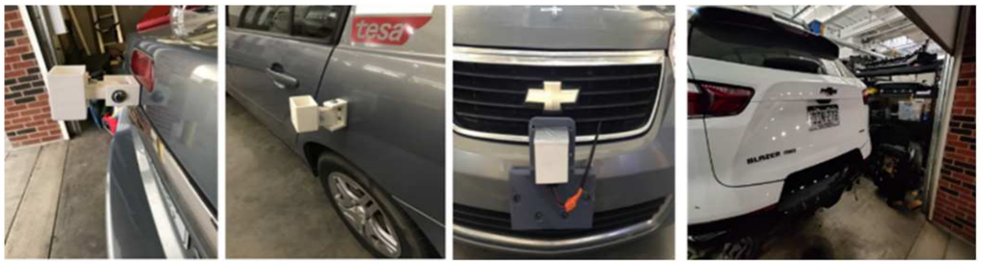


Figure 11: Sensors mounted in workspace on both car models

A Kalman filter sensor fusion algorithm was used to combine readings from sensors in a sensor configuration being evaluated, to make predictions. The longitudinal and lateral ground truth were defined for non-ego vehicles and the position error was calculated from the fused sensor measurements. The deviation of sensor measurements from ground-truth was used to calculate the values of metrics m1–m8, and hence the cost function over all test cases. Lastly, we set the stopping criterion for all three algorithms as the case when the cost function does not show a greater than 5% change over 200 iterations.

2.5.2 EXPERIMENTAL RESULTS

In our first experiment we were interested in evaluating the efficacy of different optimization algorithms (SA+GRASP, GA, and PSO) in finding optimal sensor configurations as well as exploring the consistency of the quality of solution returned by each. The cost function values for the best solution found by each algorithm for the 2016 Camaro and 2019 Blazer are shown in Figure 12. As shown in Figure 12, GA returned the solution configuration with the lowest cost function score of 0.7648 for the Camaro and 0.9252 for the Blazer. GA was able to better traverse the complex design space for our problem to arrive at the global minima compared to the SA+GRASP and PSO algorithms.

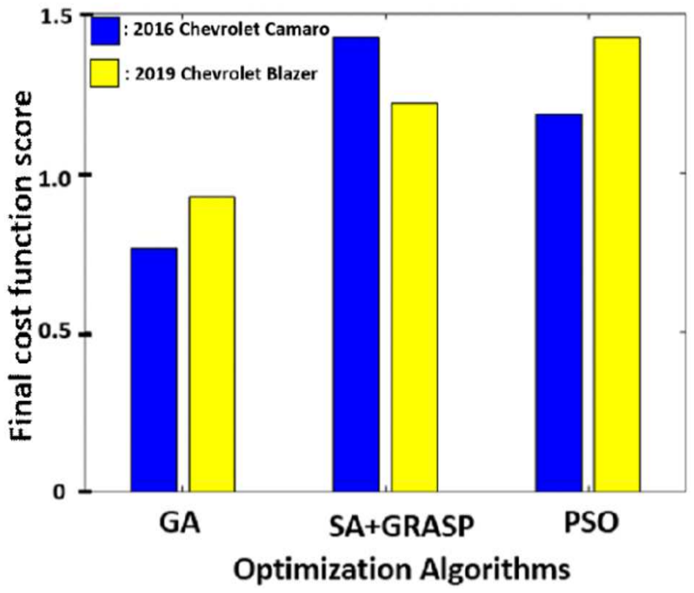


Figure 12: Cost function values for the best solution found by the SA+GRASP, GA, and PSO algorithms on the Camaro and Blazer vehicles

Next, we compared the solution generated by VESPA (utilizing the GA algorithm which gives the best results) with a baseline sensor configuration selected manually, based on best practices by a vehicle design expert in our team. This baseline configuration involved coupling a radar and camera in zones A, B, E and H each such that every mutually perpendicular direction in the 2D plane of the ego vehicle was covered using a radar and camera combined. All 8 sensors were fixed in the orientation angle, which matched the orientation of surface normal vector of the respective zone in which they were placed. The selected baseline configuration maximizes coverage by considering feature to zone correlation. This is ensured by placing at least one sensor in each region such that all zones dedicated to each of the 4 selected features is covered in the field of view of that particular sensor.

Table 5: VESPA generated solution vs baseline configuration

	VESPA Camaro	Baseline Camaro	VESPA Blazer	Baseline Blazer
Cost Function	0.9971	2.1367	1.2841	2.4630
Longitudinal position error	0.0523	0.1427	0.0845	0.2419
Lateral position error	0.1810	0.2566	0.0958	0.2204
Object occlusion rate	0.1331	0.2351	0.2062	0.3158
Velocity uncertainty	0.0823	0.1851	0.0474	0.2056
Rate of late detection	0.1158	0.2123	0.1578	0.2315
False positive lane detection rate	0.0142	0.1335	0.0221	0.1571
False negative lane detection rate	0.0214	0.0236	0.0393	0.0412
False positive object detection rate	0.0431	0.1283	0.0976	0.0954

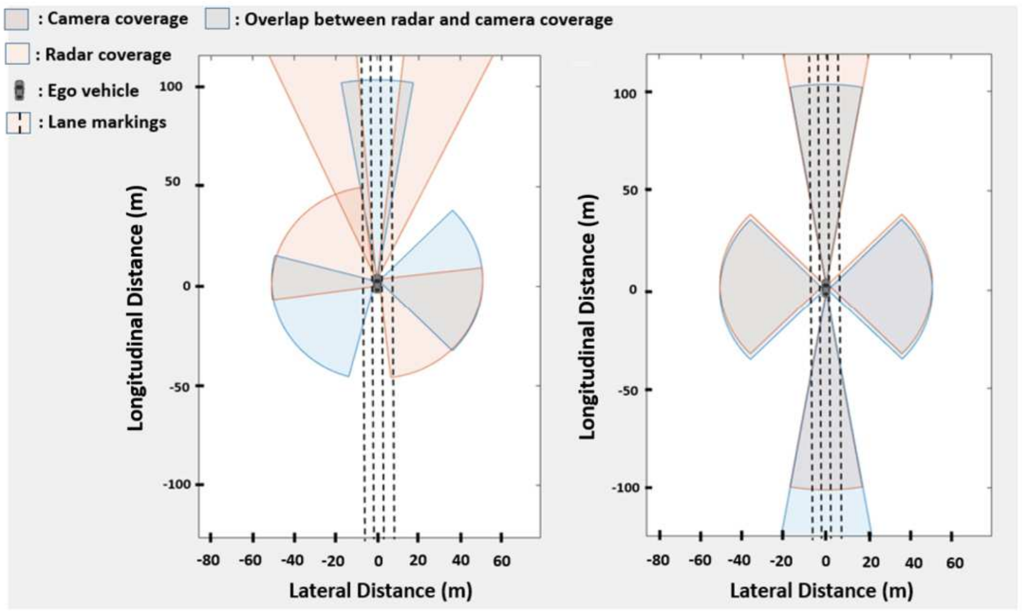
Table 5 shows the results of the comparison between the VESPA generated solution and the baseline configuration for the 2016 Camaro and 2019 Blazer. The final cost function score was higher for the baseline approach, showing that VESPA generated a significantly better (lower cost) solution for both vehicles.

Table 6: Solution from VESPA for Camaro and Blazer (in meters, degrees)

	Radar 1		Radar 2		Radar 3		Radar 4	
	VESPA Camaro	VESPA Blazer	VESPA Camaro	VESPA Blazer	VESPA Camaro	VESPA Blazer	VESPA Camaro	VESPA Blazer
X	3.7	3.7	3.7	3.65	0	2.8	3.7	2.91
Y	-0.18	-0.4	0.45	-0.9	0.9	0.9	-0.9	0.9
Z	0.15	0.2	0.20	0.2	0.2	0.25	0.2	0.2
Roll	0	0	0	0	0	0	0	0
Pitch	0	0	0	0	0	0	0	0
Yaw	15	-20	-15	-40	50	45	-130	-130
	Camera 1		Camera 2		Camera 3		Camera 4	
	VESPA Camaro	VESPA Blazer	VESPA Camaro	VESPA Blazer	VESPA Camaro	VESPA Blazer	VESPA Camaro	VESPA Blazer
X	3.7	3.7	2.8	2.8	2.8	0	X	-1
Y	0	0	-0.9	-0.9	0.9	0.9	X	0.9
Z	1.1	1.1	1.1	1.15	1.1	1.25	X	1.1
Roll	0	0	0	0	1	0	X	0
Pitch	0	0	0	0	1	1	X	1
Yaw	0	0	-90	-100	120	60	X	170

Table 6 summarizes the specific locations and orientations of the eight sensors on the two vehicles, generated by VESPA. The location and orientation information of each sensor in Table 6 is measured with respect to a global co-ordinate frame for the car model, whose origin is at the geometric center of the vehicle. An interesting observation from the table is that the sensors in the Blazer's configuration favor higher Z values than the Camaro, since the Blazer is 0.3m taller than the Camaro.

Figure 13 visualizes sensor coverage in a bird's eye plot between the best configuration generated by VESPA in figure 13(a) and the baseline configuration in figure 13(b) for the Camaro (results for Blazer are omitted for brevity). The baseline configuration was optimized with a conventional approach towards improving sensor coverage, with a secondary focus on sensor reliability.



(a) **(b)**

Figure 13: Coverage for (a): VESPA Camaro solution (b) Baseline Camaro

In contrast, the solution generated by VESPA took into account the unique strengths and weaknesses of each sensor to obtain a configuration having significantly better performance for the features supported, despite having lower overlap between field of view of different sensors than the baseline solution (figure 13) and also uses lesser number of sensors. The superiority of the VESPA solution configuration, despite using lesser number of sensors, can be accounted for by the optimized placement of camera 1, radar 2 and radar 3 in zones A and C maximizing performance of ACC and FCW. Further, in physical testing it was observed that using a radar coupled with camera in zone B for LKA reduces the number of false positives during detections. In figure 13(a), radars 3, 4 and cameras 2, 3 placed in zones D and E respectively were sufficient for improving performance of ACC and FCW by reducing the number of false positive object detections. The combined optimization of orientation and location with VESPA resulted in a sensor configuration that maximized performance for each feature.

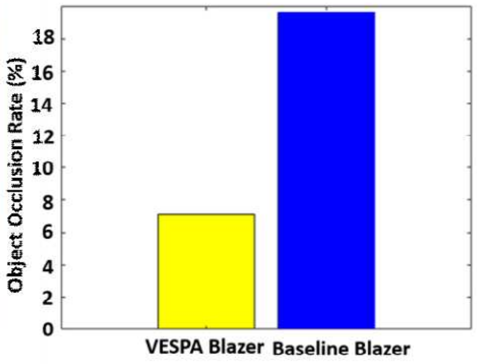


Figure 14: Performance on real drive cycle in Colorado for best solution generated by VESPA and the baseline configuration for the 2019 Blazer

Our last experiment involved testing the best sensor configuration from our VESPA framework and the baseline configuration for the 2019 Blazer on data from a real world drive cycle over one hour in Colorado. We focus only on assessing performance for the ACC and FCW features. Figure 14 shows an image from the real drive cycle with data collected by the vehicle from the radar and camera sensors on it. The figure also shows a plot of the object occlusion rate (OOR). The OOR for the baseline configuration was 19.64% (it did not detect 11 out of 56 non ego vehicles), while the VESPA generated best solution had an OOR of 7.14% (it failed to detect only 4 out of 56 non ego vehicles). The results show the effectiveness of our proposed VESPA framework in generating higher quality sensor configurations.

2.6 CONCLUSIONS

In this chapter, we propose an automated framework called VESPA that is capable of generating sensor placement and orientation in modern semi-autonomous vehicles. VESPA has the ability to optimize locations and orientations for a set of heterogeneous sensors on a given target vehicle. The framework can be tuned to improve perception on a desired collection of test cases. VESPA is also scalable across different vehicle models as shown in our analysis on the

Chevrolet Camaro and Blazer vehicles. Further, despite the sensor locations in the baseline configuration of figure 13(b) being the most intuitive, the best configuration is the one generated by VESPA, showing that even people skilled in the art of sensor placement may find it challenging to synthesize a significantly better placement than that generated by VESPA. We also validated VESPA with real drive cycle data to show its effectiveness for real-world scenarios.

3. PASTA: PERCEPTION ARCHITECTURE SEARCH TECHNIQUE FOR ADVANCED DRIVER ASSISTANCE SYSTEMS

In emerging semi-autonomous vehicles, accurate environmental perception with advanced driver assistance systems (ADAS) is critical to achieving safety and performance goals. Enabling robust perception for vehicles with ADAS requires solving multiple complex problems related to the selection and placement of sensors, object detection, and sensor fusion. Current methods address these problems in isolation, which leads to inefficient solutions. We present PASTA, a novel framework for global co-optimization of deep learning and sensing for ADAS-based vehicle perception. Experimental results with the Audi-TT and BMW-Minicooper vehicles show how PASTA can intelligently traverse the perception design space to find robust, vehicle-specific solutions.

3.1 MOTIVATION AND CONTRIBUTION

In 2020, it was reported that an estimated 38,680 people died in motor vehicle traffic crashes in the United States, representing an estimated increase of about 7.2 percent as compared to 2019 [73]. By eliminating the possibility of human driving errors through automation, advanced driver assistance systems (ADAS) are becoming a critical component in modern vehicles, to help save lives, improve fuel efficiency, and enhance driving comfort. ADAS systems typically involve a 4-stage pipeline involving sequential execution of functions related to *perception*, decision, control, and actuation. An incorrect understanding of the environment by the perception system can make the entire system prone to erroneous decision making, which can result in accidents due to imprecise real-time control and actuation. This motivates the need for a reliable *perception architecture* that can mitigate errors at the source of the pipeline and improve safety in emerging semi-autonomous vehicles. The capabilities of a perception architecture for a vehicle depend on the SAE autonomy level (defined by the SAE-J3016 standard) supported by the vehicle. In general, an optimal vehicle perception architecture should consist of carefully defined location and orientation of each sensor selected from a heterogeneous suite of sensors (e.g., cameras, radars) to maximize environmental coverage in the combined field of view obtained from the sensors. In addition to ensuring accurate sensing via appropriate sensor placement, a high object detection rate and low false positive detection rate needs to be maintained using efficient deep learning-based object detection and sensor fusion techniques. State-of-the-art deep learning based object detection models are built with different network architectures, uncertainty modeling approaches, and test datasets over a wide range of evaluation metrics [74]. For real-time perception, object detectors are resource-constrained by latency requirements, onboard memory capacity, and computational complexity. Optimizations performed to meet any one of these constraints often results in a trade-off with the performance of others [75]. As a result, comparison and selection from among the best set of deep learning based object detectors for perception applications remains a challenge. In real-world driving scenarios, the position of obstacles and traffic are highly dynamic, so after detection of an object, tracking is necessary to predict its new position. Due to noise from various sources there is an inherent uncertainty associated with the measured position and velocity. This uncertainty is minimized by using sensor fusion algorithms [76]. An important challenge with sensor fusion algorithms is that the complexity of tracking objects increases as the objects get closer, due to a much lower margin for error (uncertainty) in the vicinity of the vehicle. As summarized in figure 15, the design space of a vehicular perception architecture involves determining appropriate sensor selection and placement, object detection algorithms, and sensor fusion techniques. The possible configurations for each of these decisions is non-trivial and can easily lead to a combinatorial explosion of the design space, making exhaustive exploration impractical. Conversely, an optimization of each of these decisions individually before composing a final solution can lead to solutions that are sub-optimal and perform poorly in real environments. Today there are no generalized rules for the synthesis of perception architectures for vehicle ADAS, because perception architecture design depends heavily on the target features and use cases to be supported in the vehicle, which makes the already massive design space involved with the problem even larger and harder to traverse.

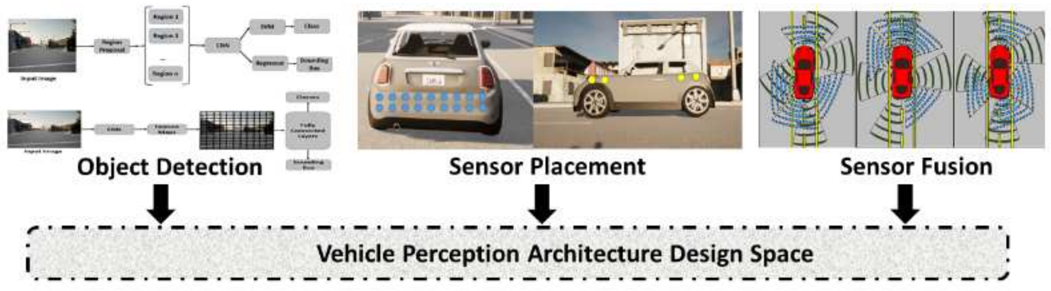


Figure 15: Breakdown of perception architecture design space

In this chapter, we propose a novel framework called PASTA (Perception Architecture Search Technique for ADAS) to perform perception architecture synthesis for emerging semi-autonomous vehicles. To the best of our knowledge, this is the first work to comprehensively explore and synthesize the sensing, fusion, and object detection perception subsystems jointly. Our experimental results indicate that the proposed framework is able to optimize perception performance across multiple ADAS metrics, for different vehicle types.

The main contributions in this chapter include:

- A global co-optimization framework capable of synthesizing robust vehicle-specific perception architecture solutions that include heterogeneous sensor placement, deep learning based object detector design, and sensor fusion algorithm selection;

- An exploration of various design space search algorithms tuned for the vehicle perception architecture search problem;
- A fast and efficient method for co-exploration of the deep learning object detector hyperparameters, through adaptive and iterative environment- and vehicle-specific transfer learning;
- A comparative analysis of the framework efficiency across different vehicle models (Audi TT, BMW Minicooper).

3.2 RELATED WORK

State-of-the-art semi-autonomous vehicles require robust perception of their environment, for which the choice of sensor placement, object detection algorithms, and sensor fusion techniques are the most important decisions. These decisions are carefully curated to support ADAS features (e.g., blindspot warning, lane keep assist) that characterize the autonomy level to be supported by a vehicle under design. Many prior works have explored vehicle perception system design with different combinations of sensor types to overcome limitations that plague individual sensor types. The work in [77] used a single camera-radar pair for perception of headway distance using a Continental radar mounted on the geometric center of the front bumper and a Nextbase 512G monocular camera behind the windscreen. Vehicle detection was performed on the collected camera frames, by sorting potential candidates in a fixed trapezoidal region of interest in the horizontal plane. In [78] a camera-radar fusion based perception architecture was proposed for target acquisition with the well-known SSD (Single Shot Detection) object detector on consecutive camera frames. This allowed their perception system to differentiate vehicles from pedestrians in real time. The detection accuracy was optimized with the use of a Kalman filter and Bayesian estimation, which reduced computational complexity compared to [77].

In [78] a single neural network was used for fusion of all camera and radar detections. The proposed neural fusion model (CRF-Net) used an optimized training strategy similar to the 'Dropout' technique, where all input neurons for the camera data are simultaneously deactivated in random training steps, forcing the network to rely more on the radar data. The training focus towards radar overcame the bias introduced by starting with pre-trained weights from the feature extractor that was trained from the camera data. The work in [79] optimized merging camera detection with LiDAR processing. An efficient clustering technique inspired by the DBSCAN algorithm allowed for a better exploitation of features from the raw LiDAR point cloud. A fusion scheme was then used to sequentially merge the 2D detections made by a YOLOv3 object detector using cylindrical projection with the detections made from clustered LiDAR point cloud data. In [80], an approach to fuse LiDAR and stereo camera data was proposed, with a post-processing method for accurate depth estimation based on a patch-wise depth correction approach. In contrast to the cylindrical projection of 2D detections in [79], the work in [80] uses a projection of 3D LiDAR points into the camera image frame instead, which upsamples the projection image, creating a more dense depth map.

All of the prior works discussed above optimize vehicle perception performance for rigid combinations of sensors and object detectors, without any design space exploration. Only a few prior works have (partially) explored the design space of sensors and object detectors for vehicle perception. An approach for optimal positioning and calibration of a three LiDAR system was proposed in [81]. The approach used a neural network to learn and qualify the effectiveness of different LiDAR location and orientations. The work in [82] proposed a sensor selection and exploration approach based on factor graphs during multi-sensor fusion. The work in [83] heuristically explored a subset of backbone networks in the Faster R-CNN object detector for perception systems in vehicles. The work in [84] presented a framework that used a genetic algorithm to optimize sensor orientations and placements in vehicles.

Unlike prior works that fine-tune specific perception architectures, e.g., [78]-[82], or explore the sensing and object detector configurations separately, e.g. [83]-[84], this chapter proposes a holistic framework that jointly co-optimizes heterogeneous sensor placement, object detection algorithms, and sensor fusion techniques. To the best of our knowledge, this is the first effort that performs co-optimization across such a comprehensive decision space to optimize ADAS perception, with the ability to be tuned and deployed across multiple vehicle types.

3.3 BACKGROUND

In this chapter, our exploration of perception architectures on a vehicle, henceforth referred to as an *ego vehicle*, targets four ADAS features that have varying degrees of longitudinal (i.e., in the same lane as the ego vehicle) and lateral (i.e., in neighboring lanes to the ego vehicle lane) sensing requirements.

3.3.1 ADAS LEVEL 2 AUTONOMY FEATURES

The SAE-J3016 standard [44] defines adaptive cruise control (ACC) and lane keep assist (LKA) individually as level 1 features, as they only perform the dynamic driving task in either the latitudinal or longitudinal direction of the vehicle. Forward collision warning (FCW) and blindspot warning (BW) are defined in SAE-J3016 as level 0 active safety systems, as they only enhance the performance of the driver without performing any portion of the dynamic driving task. However, when all four features are combined, the system can be described as a level 2 autonomy system. Figure 16 shows an overview of the four features we focus on for level 2 autonomy, which are discussed next.

Although implementations differ, all **ACC** (adaptive cruise control) systems take over longitudinal control from the driver (figure 16). The challenge in ACC is to maintain an accurate track of the lead vehicle (immediately ahead of the ego vehicle in the same lane) with a forward facing sensor and using longitudinal control to maintain the specified distance while maintaining driver comfort (e.g., avoiding sudden velocity changes).

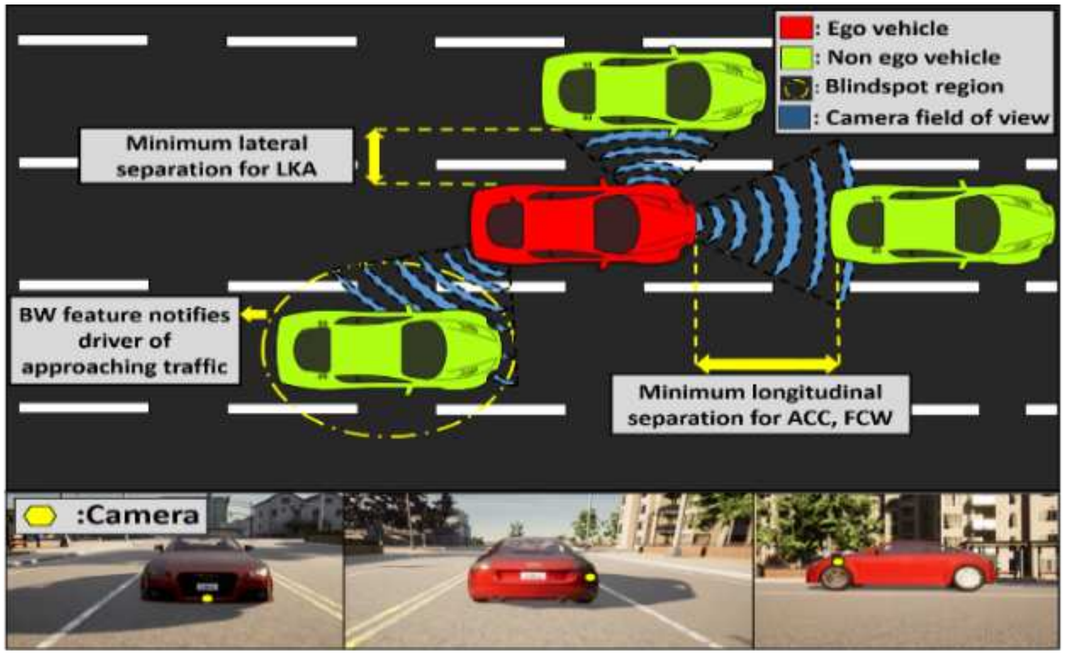

🟥: Ego vehicle
🟩: Non ego vehicle
: Blindspot region
: Camera field of view
Minimum lateral separation for LKA
BW feature notifies driver of approaching traffic
Minimum longitudinal separation for ACC, FCW
:Camera

Figure 16: Visualization of common scenarios in ACC, FCW, LKA, and BW

LKA (lane keep assist) systems determine whether the ego vehicle is drifting towards any lane boundaries and are an evolution of lane departure warning systems. LKA systems have been known to over-compensate, creating a “ping-pong” effect where the vehicle oscillates back and forth between the lane lines [85]. The main challenges in LKA are to reduce this ping-pong effect and the accurate detection of lane lines on obscured (e.g., snow covered) roads. **FCW** (forward collision warning) systems are used for real-time prediction of collisions with a lead vehicle. It is important that this system avoids false positives as well as false negatives to improve driver comfort, safety, and reduce rear end accidents [86]. Lastly, **BW** (blindspot warning) systems use lateral sensor data to determine whether there is a vehicle towards the rear on either side of the ego vehicle (figure 16) in a location the driver cannot see with their side mirrors. *A perception architecture designed to support Level 2 autonomy in a vehicle should support all four of these critical features.*

3.3.2 SENSOR PLACEMENT AND ORIENTATION

For capturing the most relevant data pertaining to each feature, sensors need to be placed strategically on the ego vehicle, such that their chosen position and orientations maximize coverage (of the vehicle environment) needed for a feature. Figure 16 shows an example of field of view coverage (in blue) corresponding to three unique placements of camera sensors on the body of the ego vehicle (in yellow, lower images) to meet coverage goals. For the ACC and FCW features, the ego vehicle is responsible for slowing down to maintain a minimum separation between the ego and lead vehicle. The camera must be positioned somewhere on the front bumper to measure minimum longitudinal separation accurately while keeping the lead vehicle in the desired field of view. For LKA, there is a need to maintain a safe minimum lateral distance between non-ego vehicles in neighboring lanes. Here a front camera is needed to extract lane line information, while side cameras are required for tracking this minimum lateral separation. As BW requires information about a specific area near the rear of the vehicle, it is a challenge to find an optimal sensor placement that maximizes the view of the blind spot. If the sensor is too far forward or too far back, it will miss key portions of the blind spots. Beyond placement, the orientation of sensors can also significantly impact coverage for all features [84]. Thus sensor placement and orientation remains a challenging problem.

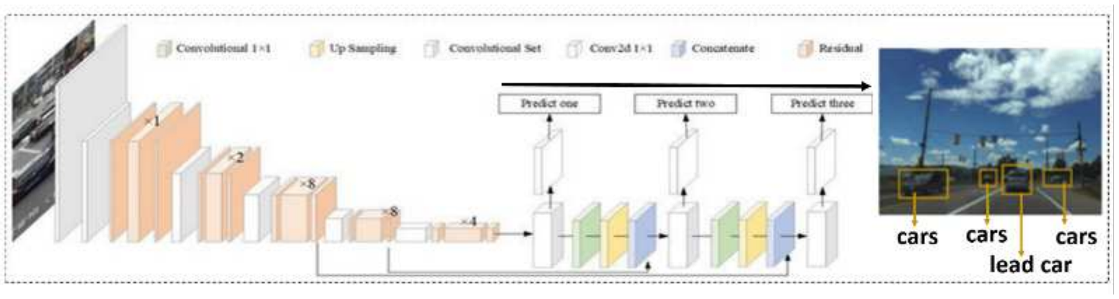


Figure 17: Example of vehicle (object) detection with YOLOv3

3.3.3 OBJECT DETECTION FOR VEHICLE ENVIRONMENT PERCEPTION

There are two broad goals associated with deep learning based object detectors: determining spatial information (relative position of an object in the image) via *localization* followed by identifying which category that object instance belongs to via *classification* [87]. As an example, Figure 17 shows object detection of multiple car instances (using the YOLOv3 deep learning based object detector [88]) by creating a bounding box around the 'car' object instances and predicting the object class as 'car'. The pipeline of traditional object detection models can be divided into informative region selection, feature extraction, and classification [89]. Depending on which subset of these steps are used to process an input image frame, object detectors are classified as single-stage or two-stage.

Modern single-stage detectors are typically composed of a feed-forward fully convolutional network that outputs object classification probabilities and box offsets (w.r.t. pre-defined anchor/bounding boxes) at each spatial position. The YOLO family of object detectors is a popular example of single-stage detectors [88]. SSD (single shot detection) is another example, based on the VGG-16 backbone [89]. An advantageous property of single-stage detectors is their very high detection throughput (e.g., ~40 frames per second with YOLO) that makes them suitable for real time scenarios [90]. Two-stage detectors divide the detection process into separate region proposal and classification stages. In the first stage, several regions in an image that have a high probability to contain an object are identified with a region proposal network (RPN). In the second stage, proposals of identified regions are fed into convolutional networks for classification. Region-based CNN (R-CNN) is an example of a two-stage detector [91]. R-CNN divides an input image into 2000 regions generated through a selective search algorithm, after which the selected regions are fed to a CNN for feature extraction followed by a Support Vector Machine (SVM) for classification. Fast R-CNN [92] and subsequently Faster R-CNN [93] improved the speed of training as well as detection accuracy compared to R-CNN by streamlining the stages.

Two-stage detectors have high localization and object recognition accuracy, whereas one-stage detectors achieve higher inference speed [94]. *In this chapter, we considered both types of object detectors to exploit the latency/accuracy tradeoffs during perception architecture synthesis.*

3.3.4 SENSOR FUSION

Perception architectures that use multiple sensors often must deal with errors due to imprecise measurements from one or more of the sensors. Conversely, errors can also arise when only a single sensor is used due to measurement uncertainties from insufficient spatial *(occlusion)* or temporal (*delayed sensor response time*) coverage of the environment. The Kalman filter is one of the most widely used sensor fusion state estimation algorithms that enables error-resilient tracking of targets [95]. The Kalman filter family is a set of recursive mathematical equations that provides an efficient computational solution of the least-squares method for estimation. The filters in this family have the ability to obtain optimal statistical estimations when the system state is described as a linear model and the error can be modeled as Gaussian noise. If the system state is represented as a nonlinear dynamic model as opposed to a linear model, a modified version of the Kalman filter known as the Extended Kalman Filter (EKF) can be used, which provides an optimal approach for implementing nonlinear recursive filters [96]. However, the computation of the Jacobian (matrix describing the system state) in EKF can be computationally expensive. Further, any attempts to reduce the cost through techniques like linearization makes the performance unstable [97]. The unscented Kalman filter (UKF) is another alternative that has the desirable property of being more amenable to parallel implementation [98]. *In our perception architecture exploration, we explore the family of Kalman filters as candidates for sensor fusion.*

3.4 PASTA ARCHITECTURE

The following section describes the proposed PASTA framework in detail.

3.4.1 OVERVIEW

Figure 18 presents a high-level overview of our proposed *PASTA* framework. The heterogeneous sensors, object detection model library, sensor fusion algorithm library, and physical dimensions of the vehicle model are inputs to the framework. An algorithmic design space exploration is used to generate a perception architecture solution which is subsequently evaluated based on a cumulative score from performance metrics relevant to the ADAS autonomy level being targeted. We evaluate three design space search exploration algorithms as part of the framework: genetic algorithm (GA), differential evolution (DE), and the firefly algorthm (FA). The process of perception architecture generation and evaluation iterates until an algorithm-specific stopping criteria is met, at which point the best design points are output. The following subsections describe each component of our framework in detail.

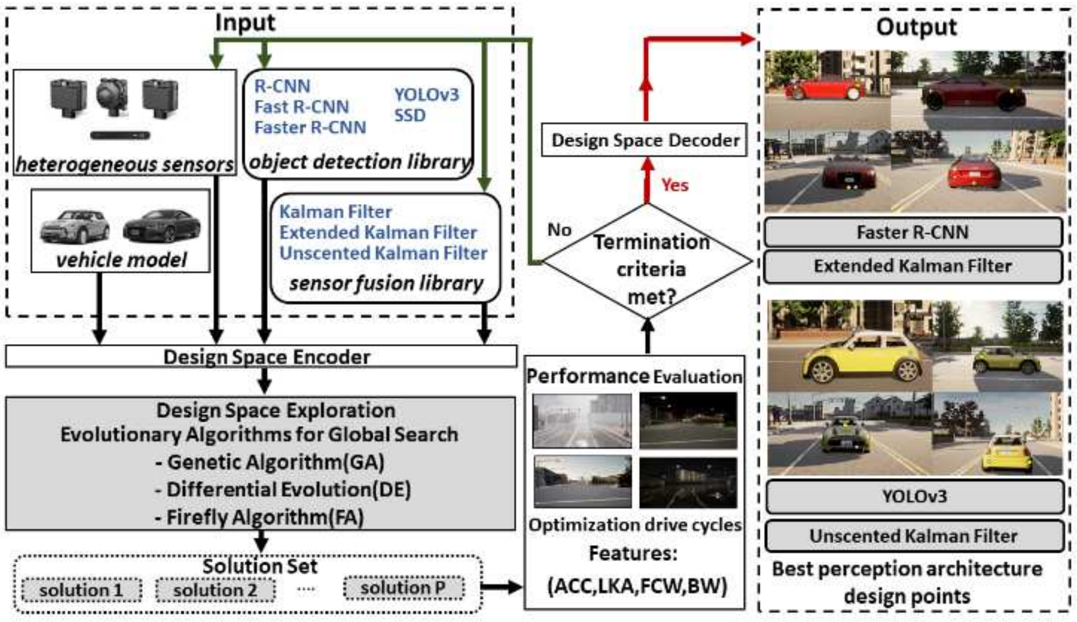


Figure 18: An overview of the proposed PASTA framework

3.4.2 PROBLEM FORMULATION AND METRICS

In our framework, for a given vehicle, we define a *design point* as a perception architecture that is a combination of three components: a *sensor configuration* which involves the fixed deployment position and orientation of each sensor selected for the vehicle, an *object detector algorithm*, and a *sensor fusion algorithm*. The goal is to find an optimal design point for the given vehicle that minimizes the cumulative error across eight metrics that are characteristic of the ability to track and detect non-ego vehicles across road geometries and traffic scenarios.

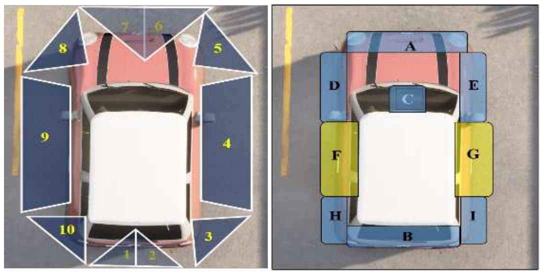
The eight selected metrics are related to our goal of supporting level 2 autonomy with the perception architecture. In the descriptions of the metrics below, the ground truth refers to the actual position of the non-ego vehicles (traffic in the environment of the ego vehicle). The metrics can be summarized as: *1) longitudinal position error and 2) lateral position error:* deviation of the detected positional data from the ground truth of non-ego vehicle positions along the y and x axes, respectively; *3) object occlusion rate:* the fraction of passing non-ego vehicles that go undetected in the vicinity of the ego vehicle; *4) velocity uncertainty:* the fraction of times that the velocity of a non-ego vehicle is measured incorrectly; *5) rate of late detection:* the fraction of the number of 'late' non-ego vehicle detections made over the total number of non-ego vehicles. Late detection is one that occurs after a non-ego vehicle crosses the minimum safe longitudinal or lateral distance, as defined by Intel RSS safety models for pre-crash scenarios [99]. This metric directly factors in the trade-off between latency and accuracy for object detector and fusion algorithms; *6) false positive lane detection rate:* the fraction of instances when a lane marker is detected but there exists no ground truth lane; *7) false negative lane detection rate:* the fraction of instances when a ground truth lane exists but is not detected; and *8) false positive object detection rate:* the fraction of total vehicle detections which were classified as non-ego vehicle detections but did not actually exist.

3.4.3 DESIGN SPACE ENCODER/DECODER

The design space encoder receives a set of random initial design points which are encoded into a vector format, best suited for various kinds of rearrangement and splitting operations during design space exploration. The encoder adapts the initial selection of inputs for our design space such that a design point is defined by the location and orientation of each sensor's configuration (consisting of six parameters: x, y, z, roll, pitch, and yaw), together with the object detector and fusion algorithm. The design space decoder converts the solutions into the same format as the input so that the output perception architecture solution(s) found can be visualized with respect to the real world co-ordinate system.

3.4.4 DESIGN SPACE EXPLORATION

The goal of a design space exploration algorithm in our framework is to generate perception architectures (design points) which are aware of feature to field of view (FOV) zone correlations around an ego vehicle. Figure 19(a) shows the 10 primary FOV zones around the ego-vehicle. These zones of interest are defined as the most important perception areas in the environment for a particular ADAS feature. Figure 19(b) shows the regions on the vehicle on which sensors can be mounted (in blue). Regions F and G (in yellow) are exempt from sensor placement due to the mechanical instability of placing sensors on the door of a vehicle. The correlation between ADAS features, zones, and regions, is shown in Figure 19(c). For exploration of possible locations within a region, a fixed step size of 2cm in two dimensions across the surface of the vehicle is considered, which generates a 2D grid of possible positions in each zone shown in Figure 19(b). The orientation exploration of each sensor involves rotation at a fixed step size of 1 degree between an upper and lower bounding limit for roll, pitch, and yaw respectively, at each of these possible positions within the 2D grid. The orientation exploration limits were chosen with caution with the caveat that some sensors, such as long range radars, have an elevated number of recorded false positives with extreme orientations.



Feature	Region	Zone
BW	B, H, I	1, 2, 3,10
LKA	E, I	3,4,5
	D, H	8, 9, 10
ACC, FCW	A, B,C	6, 7

(a) **(b)** **(c)**

Figure 19: (a) Field of view (FOV) zones; (b) sensor placement regions; (c) feature, region, and zone relationship

To get a sense of the design space, consider four sensors (e.g., two cameras and two radars). Just the determination of the optimal placement and orientation of these sensors involves exploring ${}^{1.24e+26}C_4$ and ${}^{7.34e+25}C_4$ configurations for the Audi-TT and BMW-Minicooper vehicles, respectively. Coupled with the choice of different object detectors and sensor fusion algorithms, the resulting massive design space cannot be exhaustively traversed in a practical amount of time, necessitating the use of intelligent design space search algorithms that support hill climbing to escape local minima. In our framework, we explored three evolutionary algorithms: 1) *Genetic Algorithm (GA)*, 2) *Differential Evolution (DE)*, and the 3) *Firefly Algorithm (FA)*. As shown in Figure 18, each algorithm generates a solution set of size *'P'* at every iteration until the termination criteria is met. The algorithms simultaneously co-optimize sensor configuration, object detection, and sensor fusion, and proceed to explore new regions of the design space when the termination (perception) criteria is not met. We briefly describe the three algorithms below.

3.4.4.1 GENETIC ALGORITHM (GA)

GA is a popular evolutionary algorithm that can solve optimization problems by mimicking the process of natural selection [100]. GA repeatedly selects a population of candidate solutions and then improves the solutions by modifying them. GA has the ability to optimize problems where the design space is discontinuous and also if the cost function is non differentiable. In our GA implementation, in the selection stage, the cost function values are computed for 50 design points at a time, and a roulette wheel selection method is used to select which set of chromosomes will be involved in the crossover step based on their cost function probability value (fraction of the cumulative cost function sum of all chromosomes considered in the selection). In the crossover stage, the crossover parameter is set to 0.5, allowing half of the 50 chromosomes to produce offspring. The mutation parameter is set to 0.2 which determines the new genes allowed for mutation in each iteration.

3.4.4.2 DIFFERENTIAL EVOLUTION (DE)

Differential Evolution (DE) [101] is another stochastic population-based evolutionary algorithm that takes a unique approach to mutation and recombination. An initial solution population of fixed size is selected randomly, and each solution undergoes mutation and then recombination operations. DE generates new parameter vectors by adding the weighted difference between two population vectors to a third vector to achieve difference vector-based mutation. Next, crossover is performed, where the mutated vector's parameters are mixed with the parameters of another predetermined vector, the target vector, to yield a trial vector. If the trial vector yields a lower cost function value than the target vector, the trial vector replaces the target vector in the next generation. Greedy selection is performed between the target vector and trial vector at every iteration to ensure better solutions are selected only after generation of all trial vectors. Unlike GA where parents are selected based on fitness, every solution in DE takes turns to be one of the parents [102]. In our DE implementation, we set initial population size to 50 and use a crossover probability of 0.8 to select candidates participating in crossover.

3.4.4.3 FIREFLY ALGORITHM (FA)

FA is a swarm-based metaheuristic [103] that has shown superior performance compared to GA for certain problems [104]. In FA, a solution is referred to as a firefly. The algorithm mimics how fireflies interact using flashing lights (bioluminescence). The algorithm assumes that all fireflies can be attracted by any other firefly. Further, the attractiveness of a firefly is directly proportional to its brightness which depends on the fitness function value. Initially, a random solution set is generated and the fitness (brightness) of each candidate solution is measured. In the design space, a firefly is attracted to another with higher brightness (more fit solution), with brightness decreasing exponentially over distance. FA is significantly different from DE and GA, as both exploration of new solutions and exploitation of existing solutions to find better solutions is achieved using a single position update step.

3.4.5 PERFORMANCE EVALUATION

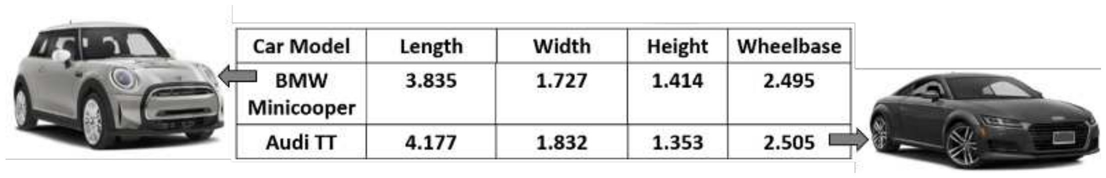
Each design point in the solution set generated per iteration of the design space exploration undergoes performance evaluation across drive cycles. A drive cycle here refers to a virtual simulation involving an ego-vehicle (with a perception architecture under evaluation) following a fixed set of waypoint co-ordinates, while performing object detection and sensor fusion on the environment and other non-ego vehicles. A total of 20 different drive cycles were considered, with 5 drive cycles customized for each ADAS feature. As an example, drive cycles for ACC and FCW involve an ego vehicle following different lead vehicles at different distances, velocities, weather conditions, and traffic profiles. The fitness of the perception architectures generated by the framework are computed using the cumulative metric scores (Section 3.4.2) across the drive cycles.

3.5 EXPERIMENTS

The following section presents the experimental setup and results involving the PASTA framework.

3.5.1 EXPERIMENTAL SETUP

To evaluate the efficacy of the PASTA framework we performed experiments in the open-source simulator CARLA (Car Learning to Act) implemented as a layer on Unreal Engine 4 (UE4) [105]. The UE4 engine provides state-of-the-art physics rendering for highly realistic driving scenarios. We leveraged this tool to design a variety of drive cycles that are roughly 5 minutes long and contain scenarios that commonly arise in real driving environments, including adverse weather conditions (rain, fog) and a few overtly aggressive/conservative driving styles observed with vehicles. To ensure generalizability, we consider a separate set of test drive cycles to evaluate solution quality, which are different from the optimization drive cycles used iteratively by the framework to generate optimized perception architecture solutions.



Car Model	Length	Width	Height	Wheelbase
BMW Minicooper	3.835	1.727	1.414	2.495
Audi TT	4.177	1.832	1.353	2.505

Figure 20: BMW Minicooper (left) and Audi TT (right)

We target generating perception architectures to meet level 2 autonomy goals for two vehicle models: Audi-TT and BMW-Minicooper (figure 20). The design space considered for evaluation uses a maximum of 4 radars and 4 cameras that can be placed in any zone (figure 19(a)-(b)). Using a greater number of these sensors led to negligible improvements for the level 2 autonomy goal. The RGB cameras possess 90^o field of view, 200 fps shutter speed, and image resolution of 800x600 pixels. The mid-range radars selected generate a maximum of 1500 measurements per second with a horizontal and vertical field of view of 30^o and a maximum detection distance of 100 meters. We considered 5 different object detectors (YOLOv3, SSD, R-CNN, Fast R-CNN, and Faster R-CNN) and 3 sensor fusion algorithms (Kalman filter, Extended Kalman filter, and Unscented Kalman filter). For the design space exploration algorithms, the cost function was a weighted sum across the eight metrics discussed in section 3.4.2, with the weight factor for each metric chosen on the basis of their total feature-wise cardinality across all zones shown in figure 19(c). During design space exploration, if the change in average cost function value was < 5% over 250 iterations, the search was terminated. All algorithmic exploration was performed on an AMD Ryzen 7 3800X 8-Core CPU desktop with an NVIDIA GeForce RTX 2080 Ti GPU.

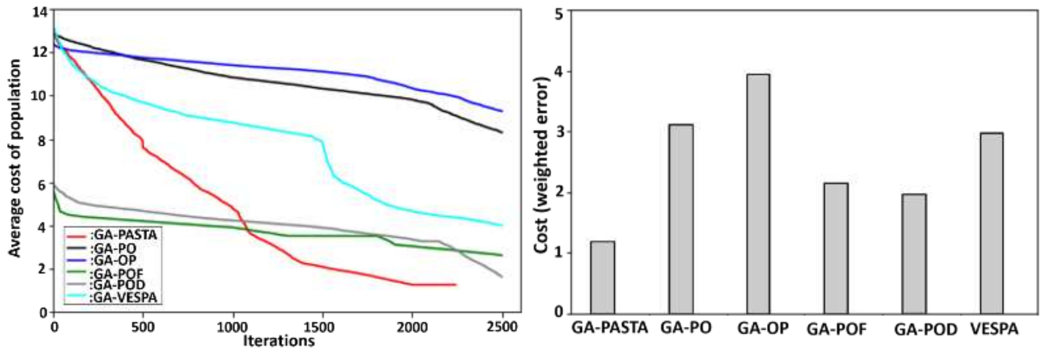
3.5.2 EXPERIMENTAL RESULTS

In the first experiment, we explored the inference latency and accuracy in terms of mean average precision (mAP) for the five different object detectors considered in this chapter. Table 7 summarizes the inference latency on a CPU and GPU, as well as the accuracy in mAP for the object detectors on images from our analyzed drive cycles, with all detectors trained on the MS-COCO dataset. It can be observed that the two-stage detectors (R-CNN, Fast R-CNN, and Faster R-CNN have a higher accuracy than the single stage detectors (SSD, YOLOv3). However, the inference time for the two-stage detector is significantly higher than for the single stage detectors. For real-time object detection in vehicles, it is crucial to be able to detect objects with low latency, typically less than 100ms [106]. As a result, single stage detectors are preferable, with YOLOv3 achieving slightly better accuracy and lower inference time than SSD. However, in some scenarios, delayed detection can still be better than not detecting or wrongly detecting an object (e.g., slightly late blindspot warning is still better than receiving no warning) in which case the slower but more accurate two-stage detectors may still be preferable.

Table 7: Object detector latency and accuracy comparison

Object detector	Latency GPU (ms)	Latency CPU (ms)	mAP (%)
R-CNN	48956.18	66090.83	73.86
Fast R-CNN	1834.71	2365.86	76.81
Faster R-CNN	176.99	286.72	79.63
SSD	53.25	70.32	70.58
YOLOv3	24.03	32.92	71.86

In PASTA, we therefore explore both single-stage and two-stage detectors, and factor in the accuracy and rate of late detection in our metrics (Section 3.4.2). Also, detectors with a higher mAP value sometimes did not detect objects that other detectors with a lower mAP were able to; thus we consider all five detectors in our exploration. Next, we explored the importance of global co-optimization for our problem. We select the genetic algorithm (GA) variant of our framework to explore the entire design space (GA-PASTA) and compared it against five other frameworks. Frameworks GA-PO and GA-OP use the GA but perform a local (sequential) search for sensor design. In GA-PO, sensor position is explored before orientation, while in GA-OP the orientation for fixed sensor locations (based on industry best practices) is explored before adjusting sensor positions. For both frameworks, the object detector used was fixed to YOLOv3 due to its sub-100ms inference latency and reasonable accuracy, while the extended Kalman filter (EKF) was used for sensor fusion due to its ability to efficiently track targets following linear or non-linear trajectories.



(a) (b)

Figure 21: (a) Comparison of perception architecture exploration frameworks; (b) Cost of best solution from each framework

The framework GA-VESPA is from prior work [55] and uses GA for exploration across sensor positions and orientations simultaneously, with the YOLOv3 object detector and EKF fusion algorithm. Frameworks GA-POD and GA-POF use GA for a more comprehensive exploration of the design space. GA-POD simultaneously explores the sensor positioning, orientation, and object detectors, with a fixed EKF fusion algorithm. GA-POF simultaneously explores the sensor positioning, orientation, and sensor fusion algorithm, with a fixed YOLOv3 fusion algorithm. Figure 21(a) depicts the average cost of solution populations (lower is better) for the BMW-Minicooper across the different frameworks plotted against the number of iterations, with each exploration lasting between 80-100 hours. It can be observed that GA-PO outperforms GA-OP, which confirms the intuitive importance of exploring sensor positioning before adjusting sensor orientations. GA-VESPA outperforms both GA-PO and GA-OP, highlighting the benefit of co-exploration of sensor position and orientation over a local sequential search approach used in GA-PO and GA-OP. GA-POD and GA-POF in turn outperform these frameworks, indicating that decisions related to object detection and sensor fusion can have a notable impact on perception quality. GA-POD terminates with its solution set having a lower average cost than GA-POF, which indicates that co-exploration of object detection and sensor placement/orientation is slightly more effective than co-exploration of sensor fusion and sensor placement/orientation. Our proposed GA-PASTA framework achieves the lowest average cost solution, highlighting the tremendous benefit that can be achieved from co-exploring sensor position/orientation, object detection, and sensor fusion algorithms. Figure 21(b) summarizes the objective function cost of the best solution found by each framework, which aligns with the population-level observations from figure 21(a).

The comparative analysis for the BMW-Minicooper was repeated three times with different initializations for all six frameworks, and the results for the other two runs show a consistent trend with the one shown in figure 21. Note also that the relative trend across frameworks observed for the Audi-TT is similar to that observed for the BMW-Minicooper, and thus the results for the Audi TT are omitted for brevity.

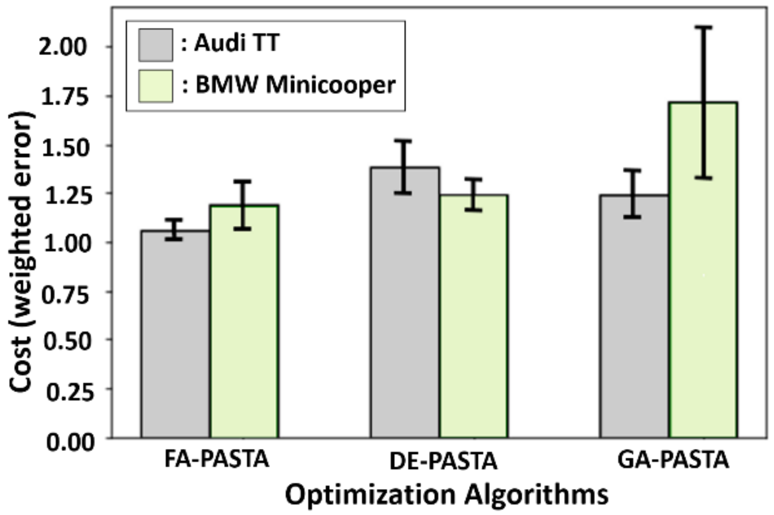
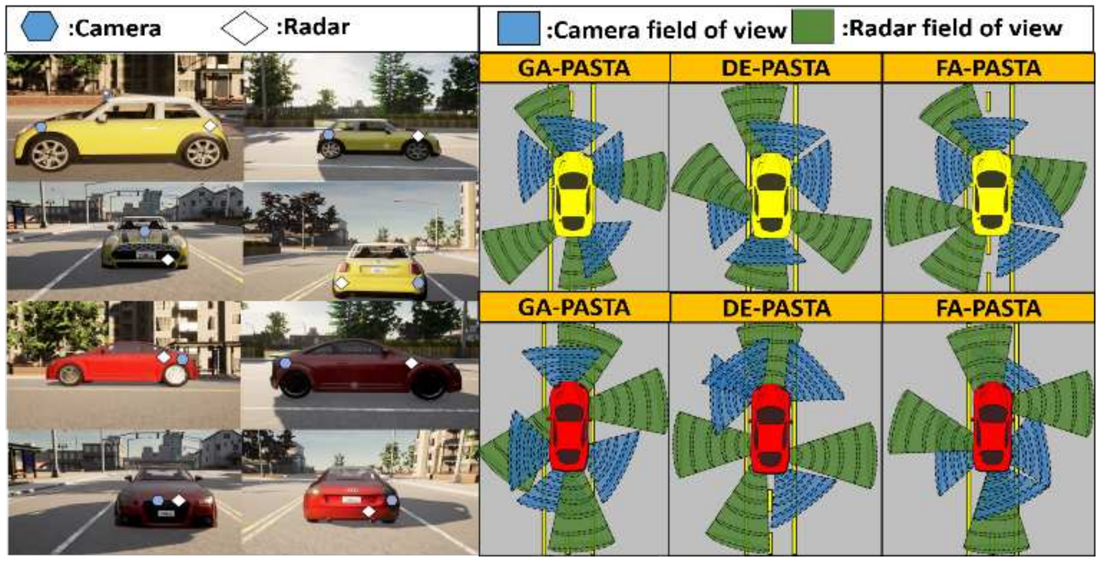


Figure 22: Comparison of three variants of PASTA framework with genetic algorithm (GA), differential evolution (DE), and Firefly algorithm (FA)

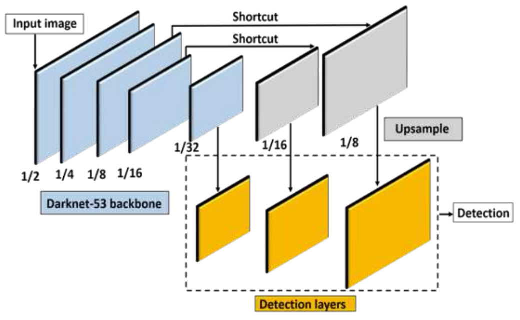
In the next experiment, we explored the efficacy of different design space exploration algorithms (GA, DE, and FA; see Section 3.4.4) to determine which algorithm can provide optimal perception architecture solutions across varying vehicle models. Figure 22 shows the results for the three variants of the PASTA framework, for the Audi-TT and BMW-Minicooper vehicles. The best solution was selected across three runs of each algorithmic variant (variations for the best solution across runs are highlighted with confidence intervals, with bars indicating the median). It can be seen that the FA algorithm outperforms the DE and GA algorithms for both vehicles. For Audi-TT, the best solution found by FA improves upon the best solution found with DE and GA by 18.34% and 14.84%, respectively. For the BMW-Minicooper the best solution found by FA outperforms the best solution found by DE and GA by 3.16% and 13.08%, respectively. Figure 23(a) depicts the specific sensor placement locations for each vehicle type, with a visualization of sensor coverage for the best solutions found by each algorithm shown in figure 23(b).

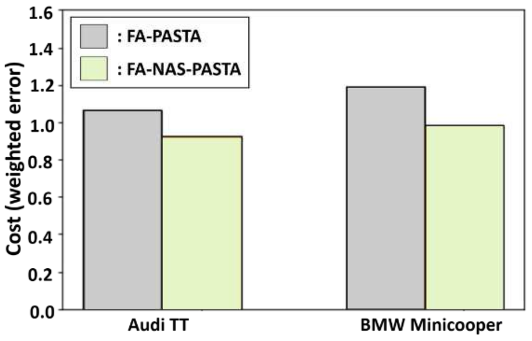


(a) **(b)**

Figure 23: (a) Sensor placement for best solution found with FA algorithm (top yellow vehicle: BMW-Minicooper, bottom red vehicle: Audi-TT); (b) Sensor coverage for best solutions found by GA, DE, and FA search algorithms

Lastly, in our quest to further improve perception architecture synthesis in PASTA, we focused on a more nuanced exploration of the object detector design space. We selected the FA search algorithm due to its superior performance over GA and DE, and modified FA-PASTA to integrate a neural architecture search (NAS) for the YOLOv3 object detector, with the aim of further improving YOLOv3 accuracy across drive cycles while maintaining its low detection latency. Our NAS for YOLOv3 involved transfer learning to retrain network layers with a dataset consisting of 6000 images obtained from the KITTI dataset, using the open source tool CADET [107]. The NAS hyperparameters that were explored involved the number of layers to unfreeze and retrain (from a total of 53 layers in the Darknet-53 backbone used in YOLOv3; Figure 24(a)), along with the optimizer learning rate, momentum, and decay. The updated variant of our framework, FA-NAS-PASTA, considered these YOLOv3 hyperparameters along with the sensor positions and orientations, and sensor fusion algorithms, during iterative evolution of the population of candidate solutions in the FA algorithm.





(a) **(b)**

Figure 24: (a) YOLOv3 object detector architecture with Darknet-53 backbone network that was fine-tuned using neural architecture search (NAS); (b) results of integrating object detector NAS with PASTA

Figure 24(b) shows the results of this analysis for the two vehicles considered. FA-PASTA is the best performing variant of our framework (from figure 22), while FA-NAS-PASTA is the modified variant that integrates NAS for YOLOv3. It can be observed that fine tuning the YOLOv3 object detector during search space exploration in FA-NAS-PASTA leads to notable improvements in the best perception architecture solution, with up to 14.43% and 21.13% improvement in performance for the Audi-TT and BMW-Minicooper, compared to PASTA-FA.

3.6 CONCLUSION

In this chapter, we propose an automated framework called *PASTA* that is capable of generating perception architecture designs for modern semi-autonomous vehicles. *PASTA* has the ability to simultaneously co-optimize locations and orientations for sensors, optimize object detectors, and select sensor fusion algorithms for a given target vehicle. Our experimental analysis showed how PASTA can synthesize optimized perception architecture solutions for the Audi TT and BMW Minicooper vehicles, while outperforming multiple semi-global exploration techniques. Integrating neural architecture search for the object detector in PASTA shows further promising improvements in solution quality.

4. CONCLUSION AND FUTURE WORK SUGGESTIONS

4.1 RESEARCH CONCLUSION

In this dissertation we identify the perception architecture design space for ADAS and address the various challenges involved. Currently, there are no formal methods or widely accepted guidelines for a top down approach to perception architecture synthesis for ADAS in academia or the automotive industry. Through our research we demonstrate that through intelligent algorithmic exploration, perception architecture solutions can be evolved as a function of the selected vehicle model.

As part of our first major contribution, we have proposed an automated framework called VESPA (Chapter 2) that is capable of generating sensor placement and orientation in modern semi-autonomous vehicles. VESPA has the ability to optimize locations and orientations for a set of heterogeneous sensors on a given target vehicle. The framework can be tuned to improve perception on a desired collection of test cases. VESPA is also scalable across different vehicle models as shown in our analysis on the Chevrolet Camaro and Blazer vehicles. Further, in our real time drive cycle testing it was observed that the VESPA generated solution reduced false positive object detection rate from 19.64% to 7.14% compared to the baseline solution.

Our second major contribution is an automated framework called PASTA (Chapter 3) that is capable of generating perception architecture designs for modern semi-autonomous vehicles. PASTA has the ability to simultaneously co-optimize locations and orientations for sensors, optimize object detectors and select sensor fusion algorithms for a given target vehicle. Our experimental analysis showed how PASTA can synthesize optimized perception architecture solutions for the Audi TT and BMW Minicooper vehicles, while outperforming multiple semi-global exploration techniques. Integrating neural architecture search for the object detector in

PASTA showed further promising improvements in solution quality. The solutions showed significant improvements with up to 14.43% and 21.13% improvement in performance for the Audi-TT and BMW-Minicooper respectively compared to the initial approach in PASTA.

4.2 SUGGESTIONS FOR FUTURE WORK

With the rapid advancement in ADAS perception architecture design methodology, the design space for sensor modality selection and placement combined with future options for object detection and sensor fusion is predicted to evolve exponentially with time. The synthesis of optimal perception architecture solutions will continue to be a more complex problem due to this growing design space and warrant constraint-aware designs with focus on reliability, safety and driver comfort. Hence, we envision the following research directions for future work:

- *Reliable hardware software co-optimization aware design of perception architecture:* Perception architecture synthesis is constrained by real time latency requirements which can be measured at the signal level. The problem can be formulated such that performance of a synthesized perception architecture is aware of which automotive grade embedded platform (like NVIDIA Drive Hyperion, Drive Orin, Arm Cortex- A65AE, Arm Cortex-A76E and Intel AIoT Tank) is selected and latency analysis can be performed at an architecture level giving important insights into optimal hardware requirements such as RAM, cache size, throughput and memory consumption that work best with object detectors and fusion algorithms.
- *Perception architecture synthesis with focus on increased driver comfort:* With integration of driver in loop (DIL) and hardware in loop (HIL) in exploration, the driver experience and safety level of the synthesized perception architecture can be evaluated. A co-optimization framework capable of quantifying driver's trust and comfort level during

navigation can help to design perception architectures which exhibit safe driving behavior and minimized traffic violations or accidents. HIL testing of the synthesized perception architecture solutions, can help to identify which embedded processes in the perception workflow are contributing to increased latency and improve reaction time of the system during difficult maneuvers enforcing driver safety.

- *A co-optimization approach for perception, control and path planning design in ADAS:* A joint co-optimization exploration of perception architecture components, trajectory planning and control system design can encompass an end to end exploration of all components critical to autonomous system design for automotive cyber physical platforms. This co-optimization framework can capture the effect of environment modeling for perception architecture synthesis on trajectory planning and control algorithm response. The performance of control algorithms and various artificial intelligence based path planning algorithms can be benchmarked against types of perception architectures to quantify the effect of error propagation in the ADAS pipeline.
- *Vehicle to Everything (V2X) fusion aware design of perception architecture for failsafe reactions in real time scenarios:* Perception architecture design is limited by single point of failure due to the high influence of individual perception architecture components. A well designed distributed computing platform with integration of V2X fusion data can ensure a failsafe reaction mechanism when perception is compromised due to limited compute resource or adverse weather conditions. V2X sensor data benefits information fusion for safe navigation since it includes critical information transferred between a given vehicle and all other moving parts of traffic and infrastructure. V2V (vehicle to vehicle) sensor data fusion can enable co-operative autonomous driving and uniform velocity

control, eliminating traffic congestions and enabling fuel efficiency. Further, V2I (vehicle to infrastructure) data fusion can be used for enhanced route planning by utilizing prior information of weather conditions and traffic density.

- *A distributed path association algorithm between ego vehicles for fuel efficiency and safety:* Path planning algorithm performance for a given ADAS design depends on the fidelity of environment and traffic modeling determined by the quality of perception. State-of-the-art path planning algorithms generate trajectories for only the ego vehicle and respond to encountered non-ego vehicles relying solely on real time sensing. An efficient path association algorithm running on all ego vehicle made available through V2X infrastructure, could improve the efficiency with which trajectories are planned and remove road congestion at low speeds or traffic accidents at high speeds guaranteeing long term fuel efficiency and safety.

BIBLIOGRAPHY

[1] NHTSA (National Highway Traffic Safety Administration), National Center for Statistics and Analysis, "Early Estimates of Motor Vehicle Traffic Fatalities and Fatality Rate by Sub-Categories in 2020", 2020

[2] U. S Department of Transportation, "USDOT Releases New Data Showing That Road Fatalities Spiked in First Half of 2021", [Online]. Available: https://www.nhtsa.gov/press-releases/usdot-releases-new-data-showing-road-fatalities-spiked-first-half-2021, 2021

[3] U. S Energy Administration, "Study of the Potential Energy Consumption Impacts of Connected and Automated Vehicles", [Online]. Available: https://www.eia.gov/analysis/studies/transportation/automated/pdf/automated_vehicles.pdf, 2017

[4] Ohio State University, "The future of driving", [Online]. Available: https://onlinemasters.ohio.edu/blog/the-future-of-driving/, 2021

[5] U. S Department of Energy, "SMART Mobility Connected and Automated Vehicles Capstone Report", [Online]. Available:https://www.energy.gov/sites/prod/files/2020/08/f77/SMART-CAVS_Capstone_07.22.20.pdf, 2021

[6] Klynveld Peat Marwick Goerdeler (KPMG), "Connected and autonomous vehicle", [Online]. Available: https://assets.kpmg/content/dam/kpmg/images/2015/05/connected-and-autonomous-vehicles.pdf, 2015

[7] A. Keoleian, A. Moorthy, R. Kleine “Autonomous Vehicles: A New Solution to the Last Mile Problem?”, *Proceedings of 8th International Society for Industrial Ecology Biennial Conference*, 2018

[8] P. Kaur, R. Sobti, "Current challenges in modelling advanced driver assistance systems: Future trends and advancements", *2nd IEEE International Conference on Intelligent Transportation Engineering (ICITE)*, 2017

[9] M. Mishra, A. Kumar, "ADAS Technology: A Review on Challenges, Legal Risk Mitigation and Solutions", *Autonomous Driving and Advanced Driver-Assistance Systems*, 2021

[10] J. Wei, J. He, Y. Zhou, K. Chen, Z. Tang, Z. Xiong, "Enhanced object detection with deep convolutional neural networks for advanced driving assistance", *IEEE Transactions on Intelligent Transportation Systems*, 2019

[11] J.P. Giacalone, L. Bourgeois, A. Ancora, "Challenges in aggregation of heterogeneous sensors for Autonomous Driving Systems", *IEEE Sensors Applications Symposium (SAS)*, 2019

[12] I. Han, D.H. Park, K.J. Kim, "A New Open-Source Off-Road Environment for Benchmark Generalization of Autonomous Driving", *IEEE Access 9*, 2021

[13] E. Marti, M.A Miguel, F. Garcia, J. Perez, "A review of sensor technologies for perception in automated driving", *IEEE Intelligent Transportation Systems Magazine*, 2019

[14] H. Moradi, A. Basireddy, "Automotive Radar Signal Analysis", *Connected and Autonomous Vehicles in Smart Cities*, 2020

[15] J. Baruah, R. Bera, S. Dhar, "Ranking of sensors for ADAS—an MCDM-based approach", *Advances in Communication, Devices and Networking*, 2018

[16] M. Warren, "Automotive LIDAR technology", *Symposium on VLSI Circuits*, 2019

[17] T. Raj, F.H. Hashim, A.B. Huddin, M.F. Ibrahim, A. Hussain, "A Survey on LiDAR Scanning Mechanisms", *Electronics*, 2020

[18] S.A Schneider, K. Saad, "Camera behavioral model and testbed setups for image-based ADAS functions", *Elektrotechnik und Informationstechnik*, 2018

[19] Z. Zhong, S. Liu, M. Mathew, A. Dubey, "Camera radar fusion for increased reliability in adas applications", *Electronic Imaging*, 2018

[20] I.J. Xique, W. Buller, Z.B. Fard, E. Dennis, B. Hart, "Evaluating complementary strengths and weaknesses of ADAS sensors", *IEEE 88th Vehicular Technology Conference*, 2018

[21] Ford Motors, "Ford, Argo AI and Walmart to launch autonomous vehicle delivery service in three U.S cities" [Online]. Available: https://media.ford.com/content/fordmedia/fna/us/en/news/2021/09/15/ford-argo-ai-and-walmart.html, 2021

[22] Nissan Motor Corporation, "ProPILOT" [Online]. Available: https://www.nissan-global.com/EN/TECHNOLOGY/OVERVIEW/propilot.html, 2018

[23] Volskwagen AG, "VW's Robo-Cars Get a Boost From Luminar's Lidar" [Online]. Available: https://www.wired.com/story/vw-audi-aid-luminar-lidar-self-driving/, 2019

[24] BMW Group, "BMW Group, Intel and Mobileye Team Up to Bring Fully Autonomous Driving to Streets by 2021" [Online]. Available: https://newsroom.intel.com/news-releases/intel-bmw-group-mobileye-autonomous-driving/, 2020

[25] Waymo, "How Google’s Autonomous Car Passed the First U.S. State Self-Driving Test" [Online]. Available: https://spectrum.ieee.org/how-googles-autonomous-car-passed-the-first-us-state-selfdriving-test, 2020

[26] Tesla Motors, "Autopilot and Full Self-Driving Capability" [Online]. Available: https://www.tesla.com/en_AE/support/autopilot-and-full-self-driving-capability, 2020

[27] Hyundai Motors, "An autonomous driving future" [Online]. Available: https://www.hyundai.com/au/en/why-hyundai/autonomous-driving, 2020

[28] X. Wu, D. Sahoo, S.C. Hoi, "Recent advances in deep learning for object detection", *Neurocomputing*, 2020

[29] S. Grigorescu, B. Trasnea, T. Cocias, G. Macesanu, "A survey of deep learning techniques for autonomous driving", *Journal of Field Robotics*, 2020

[30] J.K. Hackett, M. Shah, "Multi-sensor fusion: a perspective", *Proceedings of IEEE International Conference on Robotics and Automation*, 1990

[31] J.W. Hu, B.Y. Zheng, C. Wang, C.H. Zhao, X.L. Hou, Q. Pan, Z. Xu, "A survey on multi-sensor fusion based obstacle detection for intelligent ground vehicles in off-road environments", *Frontiers of Information Technology & Electronic Engineering*, 2020

[32] R.E Kalman, "A New Approach to Linear Filtering and Prediction Problems", *Transactions of the ASME– Journal of Basic Engineering*, 1960

[33] P.S. Maybeck, "The Kalman filter: An introduction to concepts", *Autonomous robot vehicles*, 1990

[34] F. Castanedo, "A review of data fusion techniques", *The Scientific World Journal*, 2013

[35] SAE International Standard J3016, “Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles”, 2018

[36] K. Russell, "End-to-End Learning: Using Neural Networks for Vehicle Control and Obstacle Avoidance", *Georgia Southern University Digital Commons*, 2019

[37] Mitsubishi Motors, "Company History", [Online]. Available: https://www.mitsubishicars.com/what-drives-us/history, 2019

[38] J.E Naranjo, C. González, R. García, T.D. Pedro, "Cooperative Throttle and Brake Fuzzy Control for ACC Stop & Go Maneuvers", *IEEE Transactions on Vehicular Technology*, 2007

[39] C. Bila, F. Sivrikaya, M.A. Khan, S. Albayrak, "Vehicles of the future: A survey of research on safety issues", *IEEE Transactions on Intelligent Transportation Systems*, 2016

[40] C. Kirchner, "Lane Keeping Assist Explained," Motor Review, [Online]. Available: https://motorreview.com/lane-keeping-assist-explained, 2014

[41] Consumer Reports, "Guide to Forward Collision Warning," Consumer Reports, [Online]. Available: https://www.consumerreports.org/car-safety/forwardcollision-warning-guide/, 2019

[42] National Transportation Safety Board (NTSB), "The use of forward collision avoidance systems to prevent and mitigate rear-end crashes", 2015

[43] Consumer Reports, "Guide to Blind Spot Warning," Consumer Reports, [Online]. Available: https://www.consumerreports.org/car-safety/blind-spot-warning-guide/, 2019.

[44] SAE International Standard J3016, "Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles", 2018

[45] R. Spinneker, C. Koch, S.B Park and J. J Yoon, "Fast fog detection for camera based Advanced Driver Assistance Systems", *IEEE International Conference on Intelligent Transportation Systems (ITSC),* 2014

[46] Robert Bosch GmbH, Bosch MRR Data Sheet, [Online]. Available: https://www.bosch-mobility-solutions.com, 2019

[47] C. Xu, P. Zhang, H. Wang, Y. Li, C. Li, "Ultrasonic echo waveshape features extraction based on QPSO-matching pursuit for online wear debris discrimination", *Mechanical Systems and Signal Processing*, 2015

[48] S. Li, G. Li, J. Yu, C. Liu, B. Cheng, J. Wang, K. Li, "Kalman filter based tracking of moving objects using linear ultrasonic sensor array for road vehicles", *Mechanical Systems and Signal Processing*, 2018

[49] M. Jamaluddin, A.Z. Shukor, M.F. Miskon, F.A. Ibrahim and M.Q.A. Redzuan, "An Analysis of Sensor Placement for Vehicle's Blind Spot Detection and Warning System", *Journal of Telecommunication, Electronic and Computer Engineering*, 2017

[50] T. Kim, T. Park, "Placement Optimization of Multiple Lidar Sensors for Autonomous Vehicles", *IEEE Transactions on Intelligent Transportation Systems*, 2019

[51] W. Meadows, C. Hudson, C. Goodin, L. Dabbiru, B. Powell, M. Doude, D. Carruth, M. Islam, J.E Ball, B. Tang, "Multi-LIDAR placement, calibration, co-registration, and processing on a Subaru Forester for off-road autonomous vehicles operations", *Autonomous Systems: Sensors, Processing and Security for Vehicles and Infrastructure,* 2019

[52] D. Shapiro, "Levelling Up: What is level 2 automated driving?", [Online]. Available: https://blogs.nvidia.com/blog/2019/02/06/what-is-level-2-automateddriving/, 2019

[53] J. Wenger, "Automotive radar - status and perspectives," *IEEE Compound Semiconductor Integrated Circuit Symposium*, 2005

[54] C. Kirchner, “Lane Keeping Assist Explained,” Motor Review, [Online]. Available: https://motorreview.com/lane-keeping-assist-explained, 2014.

[55] Consumer Reports, “Guide to Forward Collision Warning,” Consumer Reports, [Online]. Available: https://www.consumerreports.org/car-safety/forwardcollision-warning-guide/, 2019

[56] National Transportation Safety Board (NTSB), “The use of forward collision avoidance systems to prevent and mitigate rear-end crashes”, 2015.

[57] Consumer Reports, “Guide to Blind Spot Warning,” Consumer Reports, [Online]. Available: https://www.consumerreports.org/car-safety/blind-spotwarning-guide/, 2019.

[58] Mobileye, Intel Co., “Implementing RSS Model on NHTSA Pre-Crash scenarios”, [Online]. Available: https://www.mobileye.com/responsibility-sensitivesafety/rss_on_nhtsa.pdf, 2019

[59] A. Kumar, P. Simon, “Review of Lane Detection and tracking algorithms in Advanced Driver Assistance Systems”, *International Journal of Computer Science and Information Technology (IJCSIT)*, 2015

[60] E. Aarts, J. Korst. “Simulated Annealing and Boltzmann Machines”, *A Stochastic Approach to Combinatorial Optimization and Neural Computing*, *Wiley*, 1989

[61] O. Bohachevsky, M. E. Johnson, M. L. Stein, “Generalized Simulated Annealing for Function Optimization”, *Technometrics*, 1986

[62] P.J.M. van Laarhoven, E.H.L Aarts, “Simulated annealing: Simulated Annealing: Theory and Applications”, *Mathematics and Its Applications*, *Springer* , 1987

[63] E. Oliveira, C. H. Antunes and Á. Gomes, "A hybrid multi-objective GRASP+SA algorithm with incorporation of preferences," *IEEE Symposium on Computational Intelligence in Multi-Criteria DecisionMaking (MCDM)*, 2014

[64] D. Sosnowska, “Optimization of a simplified Fleet Assignment Problem with metaheuristics: Simulated Annealing and GRASP”, *Nonconvex Optimization and Its Applications, Springer*, 2000

[65] C. Reeves, “Genetic Algorithms: Handbook of Metaheuristics”, *International Series in Operations Research & Management Science, Springer*, 2003

[66] M. Rattray, J. L. Shapiro, “The dynamics of a genetic algorithm for a simple learning problem”, *Journal of Physics A: Mathematical and General*, 1996

[67] J. Kennedy, R. Eberhart, "Particle swarm optimization", *Proceedings of International Conference on Neural Networks (ICCN)*, 1995

[68] M. Clerc, J. Kennedy, “The particle swarm - explosion, stability, and convergence in a multidimensional complex space”, *IEEE Press*, 2002

[69] S. Huang, “A review of particle swarm optimization algorithm”, *Computer Engineering Design*, 1977.

[70] X. Lei, Z. Shi, “Application and parameter analysis of particle swarm optimization algorithm in function optimization”, *Computer Engineering Applications (CEA)*, 2008

[71] R. Kumar, S. Jayashankar, “Radar and camera sensor fusion with ROS for Autonomous Driving”, *Fifth International Conference on Image Information Processing (ICIIP)*, 2019

[72] Z. Zhong, S. Liu, M. Matthew, A. Dubey, “Camera Radar fusion for increased reliability in ADAS Applications”, *Electronic Imaging, Autonomous Vehicles and Machines*, 2018

[73] NHTSA (National Highway Traffic Safety Administration), National Center for Statistics and Analysis, "Early Estimates of Motor Vehicle Traffic Fatalities and Fatality Rate by Sub-Categories in 2020", 2020

[74] A. Gupta, A. Anpalagan, L. Guan, A.S. Khwaja , “Deep learning for object detection and scene perception in self-driving cars: Survey, challenges, and open issues,” *Array*, 2021

[75] D. Feng, A. Harakeh, S. L. Waslander and K. Dietmayer, "A Review and Comparative Study on Probabilistic Object Detection in Autonomous Driving", *IEEE Transactions on Intelligent Transportation Systems,* 2021

[76] J. Carazo, D. Rufas, E. Bimepica, J. Carrabina, "Resource-Constrained Machine Learning for ADAS: A Systematic Review", *IEEE Access*, 2020

[77] Y. Zhexiang, B. Jie, C. Sihan, H. Libo, B. Xin, "Camera-Radar Data Fusion for Target Detection via Kalman Filter and Bayesian Estimation," *SAE Technical Paper*, 2018

[78] F. Nobis, M. Geisslinger, M. Weber, J. Betz, M. Lienkamp, “A Deep Learning-based Radar and Camera Sensor Fusion Architecture for Object Detection”, *IEEE Sensor Data Fusion: Trends, Solutions, Applications (SDF),* 2020

[79] M. Verucchi, L. Bartoli, F. Bagni, F. Gatti, P. Burgio, M. Bertogna, "Real-Time clustering and LiDAR-camera fusion on embedded platforms for self-driving cars," *IEEE International Conference on Robotic Computing (IRC),* 2020

[80] L. Meng, L. Yang, G. Tang, S. Ren, W. Yang, "An Optimization of Deep Sensor Fusion Based on Generalized Intersection over Union", *International Conference on Algorithms and Architectures for Parallel Processing, Springer*, 2020

[81] W. Meadows, C. Hudson, C. Goodin, L. Dabbiru, B. Powell, M. Doude, D. Carruth, M. Islam, J.E Ball, B. Tang, “Multi-LIDAR placement, calibration, co-registration, and processing on a Subaru Forester for off-road autonomous vehicles operations”, *Autonomous Systems: Sensors, Processing and Security for Vehicles and Infrastructure*, 2019

[82] H. Chen, P. Ling, Z. Danping, L. Kun, L. Yexuan, C. Yu., "An optimal selection of sensors in multi-sensor fusion navigation with factor graph.", *Ubiquitous Positioning, Indoor Navigation and Location-Based Services (UPINLBS)*, 2018.

[83] J. Luo, H. Fang, F. Shao, Y. Zhong, X. Hua, "Multi-scale traffic vehicle detection based on faster R–CNN with NAS optimization and feature enrichment.", *Defense Technology 17*, 2021

[84] J. Dey, W. Taylor, S. Pasricha "VESPA: A Framework for Optimizing Heterogeneous Sensor Placement and Orientation for Autonomous Vehicles", *IEEE Consumer Electronics Magazine*, 2020

[85] C. Kirchner, “Lane Keeping Assist Explained,” Motor Review, [Online]. Available: https://motorreview.com/lane-keeping-assist-explained, 2014.

[86] H. Li, G. Zhao, L. Qin, H. Aizeke, X. Zhao, Y. Yang. "A Survey of Safety Warnings Under Connected Vehicle Environments", *IEEE Transactions on Intelligent Transportation Systems 22*, 2020

[87] Z. Q Zhao, P. Zheng, S.T Xu, X. Wu, "Object detection with deep learning: A review", *IEEE transactions on neural networks and learning systems*, 2019

[88] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, "You only look once: Unified, real-time object detection.", *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016

[89] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.Y Fu, A.C. Berg, "SSD: Single shot multibox detector", *European Conference on Computer Vision, Springer*, 2016

[90] J. Han, D. Zhang, G. Cheng, N. Liu, D. Xu. "Advanced deep-learning techniques for salient and category-specific object detection: a survey", *IEEE Signal Processing Magazine*, 2018

[91] R. Girshick, J. Donahue, T. Darrell, J. Malik. "Rich feature hierarchies for accurate object detection and semantic segmentation." *In Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014

[92] R. Girshick,"Fast R-CNN*", Proceedings of the IEEE International Conference on Computer Vision*, 2015

[93] S. Ren, K. He, R. Girshick, J. Sun. "Faster r-cnn: Towards real-time object detection with region proposal networks." *Advances in Neural Information Processing systems (NIPS)*, 2015

[94] J. Fayyad, M.A. Jaradat, D. Gruyer, H. Najjaran, "Deep learning sensor fusion for autonomous vehicle perception and localization: A review." *Sensors 20*, 2020

[95] R.E Kalman, "A New Approach to Linear Filtering and Prediction Problems", *Transactions of the American Society of Mechanical Engineers (ASME) –Journal of Basic Engineering*, 1960

[96] J. Simon, and J. Uhlmann. "New extension of the Kalman filter to nonlinear systems", *Signal processing, sensor fusion, and target recognition International Society for Optics and Photonics*, 1997

[97] D. Yeong, G. Hernandez, J. Barry, J. Walsh, "Sensor and sensor fusion technology in autonomous vehicles: A review." *Sensors*, 2021

[98] E. Wan, V. Merwe, "The unscented Kalman filter for nonlinear estimation", *Proceedings of the IEEE Adaptive Systems for Signal Processing, Communications, and Control Symposium*, 2000

[99] Intel, "Implementing RSS Model on NHTSA Pre-Crash scenarios", 2017

[100] C. Reeves, "Genetic Algorithms: Handbook of Metaheuristics", *International Series in Operations Research & Management Science*, 2003

[101] R. Storn, K. Price, "Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces", *Journal of Global Optimization 11*, 1997

[102] V. Kachitvichyanukul, "Comparison of three evolutionary algorithms: GA, PSO, and DE." *Industrial Engineering and Management Systems* 11, 2012

[103] X.S Yang, "Firefly algorithms for multimodal optimization," *Stochastic Algorithms: Foundations and Applications*, 2009

[104] G.D Zhou, T.H Yi, H. Zhang, H.N Li "A comparative study of genetic and firefly algorithms for sensor placement in structural health monitoring", *Shock and Vibration*, 2015

[105] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, V. Koltun. "CARLA: An open urban driving simulator", *1st Annual Conference on Robot Learning Conference on robot learning*, 2017

[106] Å. Brekke, F. Vatsendvik, F. Lindseth, "Multimodal 3d object detection from simulated pretraining", *Symposium of the Norwegian Artificial Intelligence Society*, *Springer,* 2019

[107] S. Lin, Y. Zhang, C. Hsu, M. Skach, M. Haque, L. Tang, J. Mars, "The architectural implications of autonomous driving: Constraints and acceleration", *Proceedings of the Twenty-Third International Conference on Architectural Support for Programming Languages and Operating Systems*, 2018