Architecture Design and Development of an On-board Stereo Vision System for Cooperative Automated Vehicles

Narsimlu Kemsaram[1], Anweshan Das[1], and Gijs Dubbelman[2]

Abstract— **In a cooperative automated driving scenario like platooning, the ego vehicle needs reliable and accurate perception capabilities to autonomously follow the lead vehicle. This paper presents the architecture design and development of an on-board stereo vision system for cooperative automated vehicles. The input to the proposed system is stereo image pairs. It uses three deep neural networks to detect and classify objects, lane markings, and free space boundary simultaneously in front of the ego vehicle. The rectified left and right image frames of the stereo camera are used to compute a disparity map to estimate the detected object's depth and radial distance. It also estimates the object's relative velocity, azimuth, and elevation angle with respect to the ego vehicle. It sends the perceived information to the vehicle control system and displays the perceived information in a meaningful way on the human-machine interface. The system runs on both PC (x86_64 architecture) with Nvidia GPU, and the Nvidia Drive PX 2 (aarch64 architecture) automotive-grade compute platform. It is deployed and evaluated on Renault Twizy cooperative automated driving research platform. The presented results show that the stereo vision system works in real-time and is useful for cooperative automated vehicles.**

Index Terms—Artificial intelligence, cooperative automated vehicles, deep neural network, stereo vision system.

I. Introduction

In recent years researchers have made significant progress in the field of autonomous vehicles [1] to reduce traffic congestion [2], and road accidents caused by human error [3], [4]. Autonomous vehicles do not depend on communication with other traffic but rely on multiple on-board sensors to move and navigate independently. A sensor failure or any other technical error may lead to a disastrous consequence [5]. Cooperative automated vehicles, on the other hand, can share system information with other vehicles making a significant contribution towards increasing road safety and improve mobility worldwide. Constructive information sharing will provide endless possibilities for safe driving, and multiple vehicles can collaborate to compensate for information scarcity [6]. An example of such cooperative behavior is platooning, where multiple vehicles drive together in formation [7]. In a platooning scenario, the front lead vehicle is controlled by the driver, while the ego vehicles autonomously follow the vehicle in front of it [8], [9], [10], is as shown in Figure 1. Vehicles work together in forming these platoons, coordinating with each other using a vehicle-to-vehicle (V2V) communication channel [11]. The V2V communication channel is also used by the vehicles to communicate information used to optimize the inter-vehicle distance within a platoon [12].

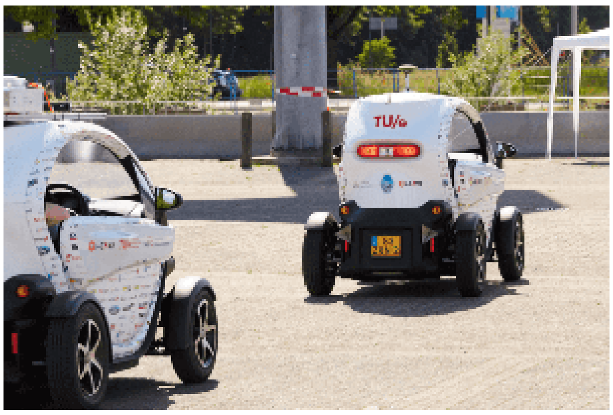


Fig. 1: Platooning scenario [10].

Automated vehicles need a robust and reliable perception system to perceive the environment accurately in real-time. Cameras [13], LiDARs [14], and Radars [15] are among the many sensors used in perception systems. We use a stereo camera for the proposed perception system for the following reasons: i) It is cost-effective as compared with other sensors such as LiDAR [16], and Radar [17], ii) Compared to Radar and LiDAR, it provides color and texture information of the surroundings, that can be used for semantic understanding of the environment, iii) It can be used to estimate the depth of objects using image disparity directly [18].

We propose an *on-board stereo vision system* for the cooperative automated vehicles. The proposed system uses the left and right images from the stereo camera to rectify and compute a disparity map, which is used to estimate the depth of the detected objects. It uses deep neural networks (DNNs) to perform object perception, lane perception, free space perception. The classified objects from the DNN are tracked in subsequent image frames using an object tracker. The system also computes the radial distance, relative velocity, azimuth, and elevation angle of the tracked objects based on motion estimation with respect to the ego vehicle. It sends the perceived obstacle information to the vehicle control system. It displays the perceived obstacle information in

[1]Narsimlu Kemsaram (`n.kemsaram@tue.nl`) and Anweshan Das (`anweshan.das@tue.nl`) are with the Department of Electrical Engineering, Mobile Perception Systems Lab, Signal Processing Systems Group, Eindhoven University of Technology (TU/e), 5612 AZ Eindhoven, The Netherlands.

[2]Gijs Dubbelman (`g.dubbelman@tue.nl, gijs@aiim.ai`) is with the Department of Electrical Engineering, Mobile Perception Systems Lab, Signal Processing Systems Group, Eindhoven University of Technology (TU/e) and AI in Motion (AIIM), 5657 EB Eindhoven, The Netherlands.

a meaningful way to the driver through the display. The schematic overview of the proposed system that is used in the cooperative automated vehicles is shown in Figure 2. The proposed system is developed on Ubuntu 16.04 (x86_64 architecture) with Nvidia GPU and deployed on Nvidia Drive PX 2 (aarch64 architecture) automotive-grade compute hardware. It is evaluated on a cooperative automated driving research platform consisting of Renault Twizy's in real-time with a custom-built automotive-grade Gigabit Multimedia Serial Link (GMSL) stereo camera.

The main contributions of this paper are:

- Developed and integrated an *on-board stereo vision system* for cooperative automated vehicles, which can detect and classify the objects, lane markings, and free space boundary in front of the vehicle.
- Developed a stereo vision-based motion estimation module, which computes the tracked object's radial distance, relative velocity, azimuth, and elevation angle.
- Evaluated and demonstrated the proposed system performance in real-time with the cooperative automated driving research platform.

The rest of this paper is structured as follows: In section II, we describe the proposed system in detail. In section III, we discuss the experiments and results. We conclude our paper in section IV.

II. Proposed On-board Stereo Vision System

We propose an independent electronic control unit of the *on-board stereo vision system* for the Integrated Cooperative Automated Vehicles (i-CAVE) project. It is a part of the i-CAVE perception system, which is used to perceive the environment in real-time. At present, the i-CAVE platoon consists of one lead vehicle and one ego vehicle. The lead vehicle is controlled by the driver, and the ego vehicle autonomously follows the lead vehicle. Our proposed system is deployed on the ego vehicle, which perceives the traffic environment with a front-facing stereo camera. It provides real-time environment information to the vehicle control system. The vehicle control system uses the perceived environment information and generates a control command to autonomously follow the lead vehicle.

A. Architecture Design

The proposed system's architecture design is divided into two parts, namely software functional architecture and hardware functional architecture, which is explained below.

1) Software Functional Architecture: The functional software architecture of the proposed system is shown in Figure 3. Raw input image frame from the stereo camera is processed and then used to perceive the vehicle's surroundings. This process is composed of the following modules, which enables a modular approach for the development of the stereo vision software: *acquisition*, *stereo vision*, *deep neural networks*, and *motion estimation*.

The *acquisition* module acquires the left and the right raw input image frame in the RCCB (Red-Clear-Clear-Blue)

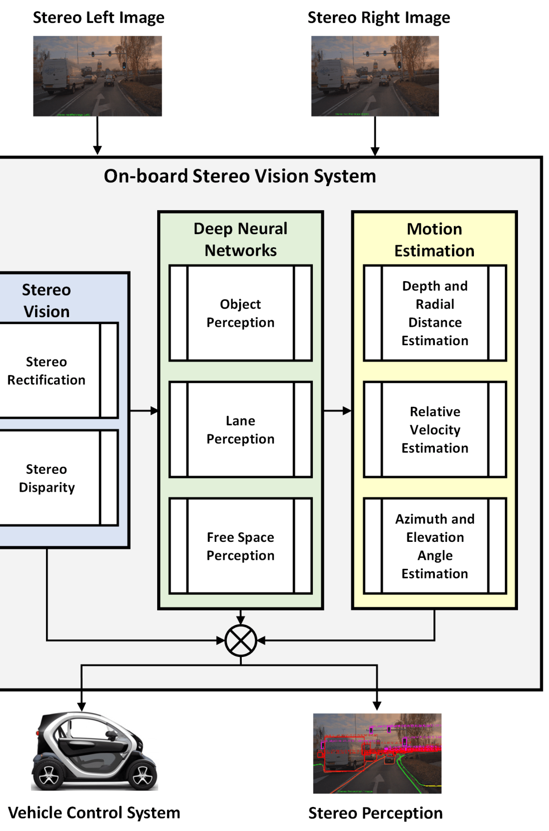


Fig. 2: Schematic overview of the proposed on-board stereo vision system for cooperative automated vehicles.

Bayer format simultaneously from the on-board stereo camera sensor. It reads the rig configuration file (JSON format) containing camera calibration parameters and converts the input image frames into the compatible RGBA (Red-Green-Blue-Alpha) image format for the *stereo vision*, *deep neural networks*, and *motion estimation* modules. The *stereo vision* module uses the camera calibration parameters acquired from the rig configuration file to undistort and rectify the input stereo image pairs. The rectified stereo image pairs are used to compute the disparity map. The *deep neural networks* module consists of three sub-modules, such as *object perception*, *lane perception*, and *free space perception*. The *object perception* sub-module classifies and tracks the detected objects and provides the region of interests (ROIs) to the *motion estimation* module. The *lane perception* sub-module identifies and classifies the lane markings as left adjacent lane, left ego lane, right ego lane, and right adjacent ego lane if they are present on the road. The outputs of this sub-module to the *motion estimation* module are identified as lane markings. The *free space perception* sub-module recognizes and classifies the drivable free space in front of the vehicle. The output of this sub-module to the *motion estimation* is the free space boundary of the identified drivable free space. The *motion estimation* module takes the tracked objects's ROIs from the *object perception* sub-module, disparity map from the *stereo disparity* sub-module. It computes the depth of each tracked object. It estimates

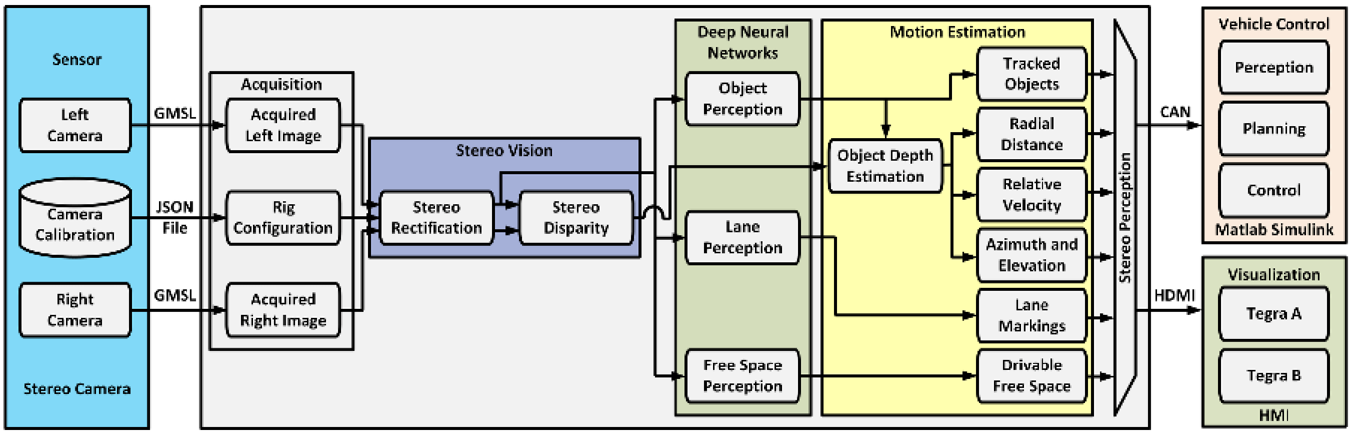


Fig. 3: Software functional architecture of the proposed on-board stereo vision system for cooperative automated vehicles.

the radial distance, relative velocity, azimuth, and elevation angle of the tracked objects. The outputs of the previous modules are sent to the vehicle control system over controller area network (CAN) and overlaid on the input image that is displayed on the human-machine interface (HMI) display over HDMI-cable.

2) Hardware Functional Architecture: The functional hardware architecture of the proposed system is shown in Figure 4. We use a dedicated Nvidia Drive PX 2 hardware platform for our research work because it is a powerful and efficient automotive-grade platform for real-time applications. The proposed system hardware consists of the following components: *GMSL stereo camera*, *power supply*, *host PC*, *Drive PX 2*, *real-time PC*, and *HMI*.

The custom-built *GMSL stereo camera* component is connected to Drive PX 2 using the Fakra Coax cable. The *power supply* component uses a DC-DC converter to convert the 24V DC input from the vehicle battery to 12V DC to power the Drive PX 2 hardware unit using an 8-pin power supply cable. The stereo vision software is developed on the *host PC*, and the compiled executable is deployed on the Drive PX 2 (target). The host is connected to the target using a Gigabit Ethernet (GbE) port, which is used to monitor and debug the target while running. The *Drive PX 2* AutoChauffeur configuration consists of two Tegra System-on-Chip (SOC), namely Tegra A (Parker A) and Tegra B (Parker B). Additionally, an ASIL-D safety based Infineon Aurix TC297 microcontroller unit (MCU) is used to configure and control the entire system. Each of the Tegra SoCs consists of a CPU with four A57 cores and two Denver2 cores, and one integrated GPU (iGPU). Each SoCs can access a PASCAL architecture based discrete GPU (dGPU) through the dedicated PCIe bus. Both the SoCs can communicate with each other via Gigabit Ethernet. The Drive PX 2 hardware supports twelve GMSL cameras using Fakra Coax cable connectors. The twelve connectors are divided into three groups (Group A, Group B, Group C), each group consisting of four ports (Port 0, 1, 2, 3). We have connected the left camera of the stereo setup to Port A0 of Group A, and the right camera to Port A0 of Group B. It supports six CAN channels, can-1 to can-4 are forwarded through Aurix microcontroller, and can-5 to can-6 are directly accessed by the Tegra A and Tegra B, respectively. The proposed system runs on a Tegra A or Tegra B of Drive PX 2 hardware, which directly communicates with the vehicle control system over the CAN-bus. The *real-time PC* runs a Simulink real-time kernel and executes the vehicle control system Simulink model. The vehicle control system model consists of three modules, namely perception, planning, and control, which directly communicates with the actuators over the vehicle CAN-bus. The *HMI* is used to display the results from stereo vision software via HDMI cable (HDMI Parker A/B port).

B. *Development*

The developed modules to classify objects, compute the radial distance, relative velocity, azimuth, and elevation angle of objects, are explained below.

1) Stereo Vision: The acquired left and right synchronized input images from the stereo camera are used for stereo rectification and disparity computation. The stereo rectification algorithm runs on the GPU and generates rectified images based on the stereo camera calibration parameters. The stereo disparity algorithm takes the rectified images and runs on the GPU using CUDA programming. It computes a disparity map using a stereo semi-global matching (SGM) algorithm [19].

2) Deep Neural Networks: The deep neural networks (DNNs) module is used to provide semantic information in front of the vehicle. It consists of three sub-modules, such as *object perception*, *lane perception*, and *free space perception*.

The *object perception* sub-module uses Nvidia's proprietary DNN called *DriveNet* [20] to perform real-time object detection and tracking. The *DriveNet* object detection algorithm runs on the CPU and GPU. It consists of three parts, such as object detection, object clustering, and object tracking. The input to the object detection algorithm is the RGBA image format, and the output is object proposals with bounding boxes. Each object can have multiple proposals. The object clustering algorithm clusters these various proposals into one bounding box for each detected object. The object tracking algorithm tracks the detected

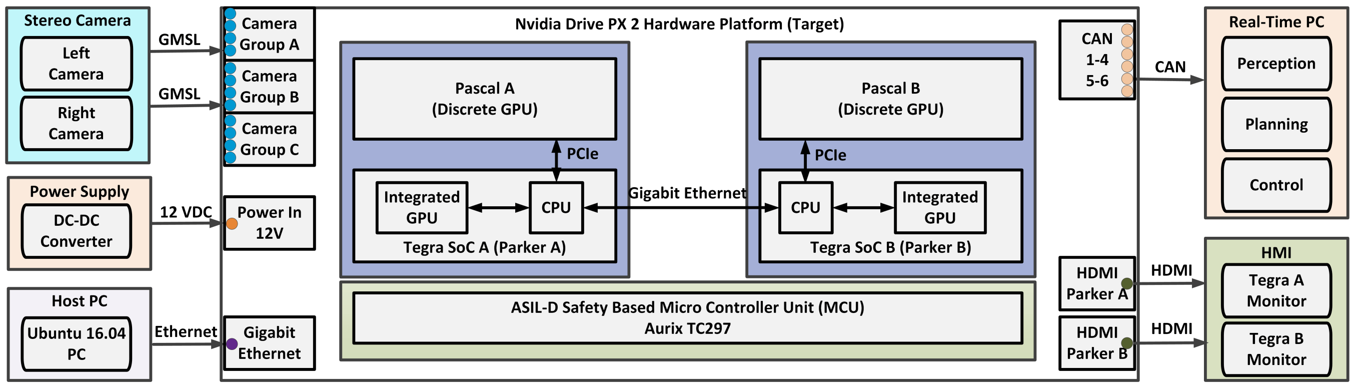


Fig. 4: Hardware functional architecture of the proposed on-board stereo vision system for cooperative automated vehicles.

bounding boxes to maintain temporal consistency. The output of this algorithm is the bounding boxes and class labels of the detected objects. It detects and classifies five different classes of objects: car, pedestrian, bicycle, traffic sign, and traffic light. The color of the bounding boxes represents the classes that it detects: red for cars, green for traffic signs, blue for bicycles, magenta for traffic lights, and orange for pedestrians. The *lane perception* sub-module is used to provide lane markings information in front of the ego vehicle. It uses Nvidia's proprietary DNN called *LaneNet* [20] to perform real-time lane detection and classification. The *LaneNet* lane detection algorithm runs on the CPU and GPU. The input image to this network is the RGBA image format, and the output is polylines representing lane markings. It calculates a probability map of lane markings for each pixel using an encoder-decoder architecture on the input image. The probability map is then binarized into clusters of lane-markings through polylines fitted to assign lane position types. The output of this algorithm is the lane markings represented as polylines and labels for the identified lane lines. It detects and classifies four different types of lane markings. The classified lane markings are left adjacent-lane, left ego-lane, right ego-lane, and right adjacent-lane when they are present on the road. The colors of the polylines represent the lane marking position types are as follows: yellow for left adjacent-lane, red for left ego-lane, green for right ego-lane, and blue for the right adjacent-lane. The *free space perception* sub-module is used to estimate the drivable free space in front of the ego vehicle. It uses Nvidia's proprietary DNN called *FreeSpaceNet* [20] to perform the real-time drivable free space boundary detection and classification. The *FreeSpaceNet* detection algorithm runs on the CPU and GPU. The input to the network is the RGBA image format, and the output is the free space boundary for the recognized drivable free space. The boundary separates the obstacle from the drivable free space. Each pixel on the boundary is associated with one of the four semantic labels: red for the vehicle, blue for a pedestrian, green for the curb, and yellow for others.

3) Motion Estimation: In this module, the radial distance, relative velocity, azimuth, and elevation angle of the tracked objects with respect to the ego vehicle are estimated. This module runs on the CPU. The depth estimation process uses a stereo camera setup in standard stereo geometry or canonical stereo or frontal parallel. The two cameras of the stereo camera setup are identical, with the same spatial resolution, the same focal length, and the parallel optical axes.

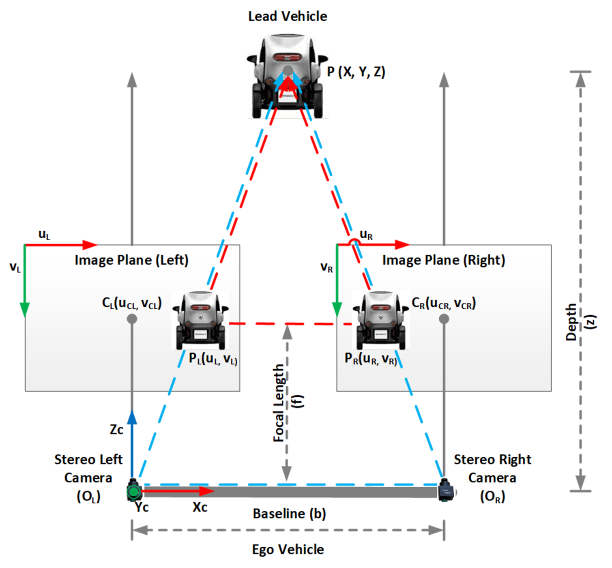


Fig. 5: A stereo vision geometric model for depth estimation. The camera coordinate system (X_c, Y_c, Z_c) is relative to the left camera's center of projection. The x-axis (X_c) points to the right of the image plane, the y-axis (Y_c) points to the bottom of the image plane, and the z-axis (Z_c) points forward, along the optical axis. The image coordinate system (u, v) is relative to the upper left corner of the image.

Depth estimation: The Figure 5 depicts the stereo triangulation technique used to estimate the depth of a lead vehicle with respect to the ego vehicle in a platooning scenario. The position of the midpoint of the lead vehicle in the camera coordinate system is $P(X, Y, Z)$ observed from both cameras at projection point $P_L(u_L, v_L)$ in the left image plane and $P_R(u_R, v_R)$ in the right image plane with u along with horizontal axis and v along with vertical axis. The principal points of both the cameras are $C_L(u_{CL}, v_{CL})$ and $C_R(u_{CR}, v_{CR})$ respectively. The left camera O_L and right camera O_R are fixed on the roof of the ego vehicle with a constant baseline distance (b). Using the similar triangles ΔPO_LO_R (blue dashed lines triangle) and ΔPP_LP_R (red dashed lines triangle), we can derive the following expression [21]:

$$z/b = (z - f)/(b - (u_L - u_R)) \tag{1}$$

from which we can obtain:

$$z/b = (z - f)/(b - d) \tag{2}$$

where, d = (u_L-u_R) is disparity in pixels, from which we can obtain:

$$z = f * b/d \tag{3}$$

where, z is the depth of the point P from the stereo camera in meters, f is the effective focal length of the stereo camera in pixels, and b is the baseline between the left and right cameras in meters. Focal length and baseline are stereo camera constants that are obtained from the stereo camera calibration.

Radial distance estimation: The Figure 6 depicts the stereo vision geometric model that is used to estimate the radial distance of a lead vehicle with respect to the ego vehicle in a platooning scenario. The lead vehicle's position $P(X, Y, Z)$ in the camera coordinate system can be calculated using the computed depth of the point P (z) in meters, focal length (f) in pixels, corresponding image point $P_L(u_L, v_L)$ in the left image plane [22]:

$$(X, Y, Z) = ((u_L - u_{CL}) * z/f, (v_L - v_{CL}) * z/f, z) \tag{4}$$

from which we can compute the radial distance (ρ) in meters:

$$\rho = \sqrt{X^2 + Y^2 + Z^2} \tag{5}$$

The radial distance is computed by taking the average of the radial distance of all points in a small region around the point P. The region is estimated by taking $1/5th$ of the size of the ROI output from the *object perception* sub-module. The standard deviation of the radial distance is also computed to estimate the noise in the distance computation.

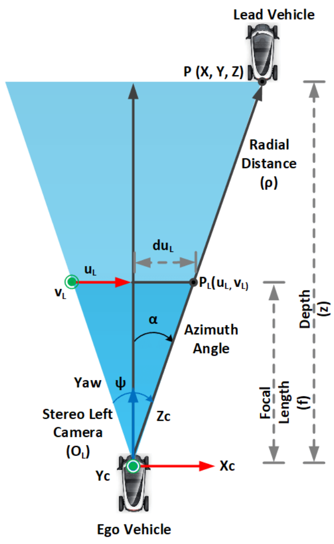


Fig. 6: A stereo vision geometric model for distance estimation (top view).

Relative velocity estimation: The Figure 7 depicts the stereo vision geometric model that is used to estimate the relative velocity of a lead vehicle with respect to the ego vehicle in a platooning scenario. The relative velocity $V_{(P,Q)}$ (in meters/seconds) of the lead vehicle in the camera coordinate system can be obtained by computing the displacement from the previous position (P) to the current position (Q) in terms of change of radial distance ($\Delta\rho$) in meters within the time (dt) in seconds:

$$V_{(P,Q)} = \frac{\Delta\rho}{dt} \tag{6}$$

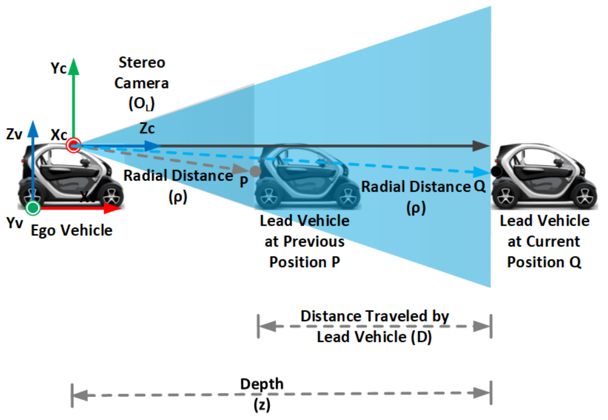

Yc
Stereo Camera (O_L)
Zv
Xc
Zc
Radial Distance (ρ)
P
Radial Distance (ρ)
Q
Yv
Ego Vehicle
Lead Vehicle at Previous Position P
Lead Vehicle at Current Position Q
Distance Traveled by Lead Vehicle (D)
Depth (z)

Fig. 7: A stereo vision geometric model for velocity estimation (side view). The computed radial distance of previous vehicle position (P) is marked with red dashed lines and current position (Q) is marked with blue dashed lines. The vehicle coordinate system (X_v, Y_v, Z_v) is relative to the center of the rear axle and on the ground. The x-axis (X_v) points forward to the front of the vehicle, the y-axis (Y_v) points to the left of the vehicle, and the z-axis (Z_v) points upwards.

Azimuth angle estimation: The Figure 6 depicts the stereo vision geometric model that is used to estimate the azimuth angle of a lead vehicle with respect to the ego vehicle in a platooning scenario. After obtaining the pixel coordinate $P_L(u_L, v_L)$ of the lead vehicle from the object tracker algorithm, and with the position of the optical center of the left image $C_L(u_{CL}, v_{CL})$, the relative horizontal distance from the optical center is (du_L) (in pixels) computed using the following expression [22]:

$$du_L = (u_L - u_{CL}) \tag{7}$$

from which we can compute the azimuth angle (α) (in degrees) of the point P using the focal length of the left camera (in pixels):

$$\alpha = arctan(du_L/f) \tag{8}$$

Elevation angle estimation: The Figure 8 depicts the stereo vision geometric model used to estimate the elevation angle of a traffic light with respect to the ego vehicle while performing platooning.

After obtaining the pixel coordinate, $P_L(u_L, v_L)$ of the traffic light from the *object tracker* algorithm, and with the position of the optical center of the left image

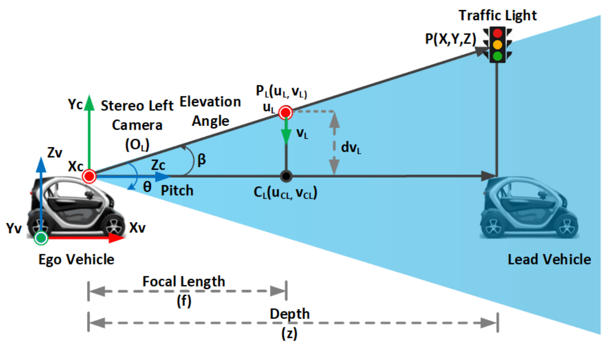


Fig. 8: A stereo vision geometric model for elevation estimation (side view).

$C_L(u_{CL}, v_{CL})$, the relative vertical distance from the optical center is (dv_L) (in pixels) computed using the following expression [22]:

$$dv_L = (v_L - v_{CL}) \tag{9}$$

from which we can compute the elevation angle (β) (in degrees) of the point P using the focal length of the left camera (f) (in pixels):

$$\beta = arctan(dv_L/f) \tag{10}$$

The above methods are developed in C++ with Nvidia DriveWorks 1.2, CUDA 9.2, and CuDNN 7.4.1 on an Ubuntu 16.04 LTS and deployed on a Drive PX 2 hardware platform.

III. Experiments and Results

In this section, we describe the experimental setup and evaluate the performance of the proposed *on-board stereo vision system*. We conducted the experiments on a Renault Twizy cooperative automated driving research platform. It consists of one lead vehicle and one ego vehicle. The ego vehicle, Renault Twizy, which is equipped with a GMSL stereo camera, Nvidia Drive PX 2 hardware platform with a stereo vision software, and 12V DC USB powered HDMI 10.1 inch small monitor, is shown in Figure 9. The Drive PX 2 requires a 12V DC power supply to operate. Hence, we added a DC-DC converter to convert from a car battery 24V DC to 12V DC. We used two identical Sekonix GMSL automotive-grade cameras (SF3325) with an ONSEMI CMOS AR0231 image sensor, 2.3 MegaPixel (1920x1208 resolution), 60 degrees horizontal field of view (ψ in Figure 6) and 30 degrees vertical field of view (θ in Figure 8), for our custom-built stereo camera setup. It also includes the mounting solution for the cameras. This consists of the aluminum beam onto which the cameras are mounted on the car roof using a rigid mounting bar at a baseline distance of 30 centimeters. The ego vehicle is also equipped with an NXP Cocoon Long-Range Radar, which we have used to compare the results of the proposed system. The Radar operates at a frequency of 78 GHz and has a horizontal field of view of 120 degrees and a vertical field of view of 20 degrees. It generates measurements at 14 Hz. The Radar is rigidly fixed in the middle of the front bumper of the ego vehicle. We used the Kvaser Leaf Light HS V2 CAN monitor tool to read the CAN messages of the proposed system in lab testing. The CAN message from the proposed system (Drive PX 2), which corresponds to the object information (class type, radial distance, relative velocity, azimuth, and elevation angle) are decoded using the database container (dbc) file by the real-time control system and generates a control command. The experiments are performed in two different scenarios: a) Controlled outdoor environment, where the position and movement of objects in the environment are known, we compare the measurements of the proposed system with the Radar b) Traffic environment, where we discuss the output of the DNNs in the proposed system on a highway. The execution time of the proposed system in two different platforms are also discussed.

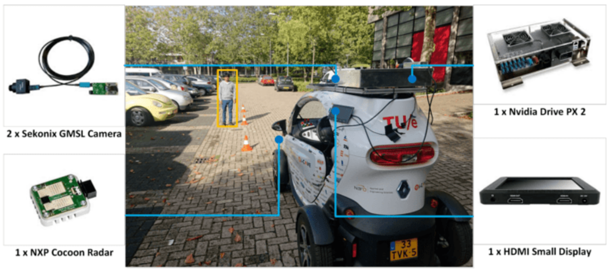
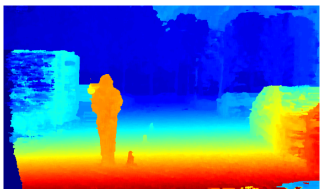


Fig. 9: The experimental setup of the Twizy cooperative automated research vehicle platform.

A. *Controlled Outdoor Environment*

In the scenario depicted in Figure 10, the vehicles were stationary, and a person moved away from the ego vehicle. The person started from 3 meters away from the ego vehicle, then moved to 6 meters, 9 meters, and ultimately to 12 meters in a period of 25 seconds. The output of the results from the proposed system is plotted and compared with the measurements of the Radar. The Figure 11a shows the plot of the measured radial distance (in meters) of all objects detected by the Radar, as this Radar cannot classify an object. We can see that the distance of most of the objects is constant except one, which starts from 3 meters and ultimately stops at 12 meters. The curve flattens and increases after every 3 meters because the person remained stationary for some seconds after moving every 3 meters. The multiple of 3 meters distance is marked with cones, as shown in Figure 10b. The Figure 11b shows the plot of the measured radial distance (in meters) of the tracked person by the proposed system. We can see a similar trend in the curve where it flattens and increases after every 3 meters as compared to the Radar measurements. The Figure 11c shows the plot of the measured radial distance (in meters) of the tracked vehicles by the proposed system. We can see that the measured distance of the vehicles is constant as expected. The Figure 12a shows the plot of the measured relative velocity (meters/seconds) of all objects detected by the Radar. We can see a clear increase and decrease in velocity when the person moves and stops, respectively. The Figure 12b shows the plot of the measured relative velocity (in meters/seconds) of the tracked person by the proposed system. We can see a similar trend in the curve, and the measurements are much noisier than the Radar system. The noise can be reduced by using temporal filters. The Figure 12c shows the plot of the measured relative velocity (in meters/seconds) of the tracked vehicles by the proposed system. The measurements are noisy but are always close to zero. The noisy measurements are present because of erroneous disparity map estimation due to improper stereo-image synchronization or error stereo matching algorithms. The Figure 13a shows the plot of the measured azimuth (in degrees) of all detected objects detected by the Radar. The azimuth of the detected objects is noisy but tends to be constant. The Figure 13b shows the plot of the measured azimuth (in degrees) of the tracked person, and Figure 13c shows the plot of the measured azimuth (in degrees) of the tracked vehicles by the proposed system. We can see that the proposed system's measurements are less noisy than the Radar. There is a small difference in the plot of the vehicle's azimuth because some vehicles were missed by the Radar. The Figure 14a shows the plot of the measured elevation (in degrees) of all detected objects detected by the Radar. The elevation does not vary that much as all the vehicles are stationary, and the change in elevation of the person moving away from the vehicle is small as person always stays close to the center of the image. The Figure 14b shows the plot of the measured elevation (in degrees) of the tracked person, and Figure 14c shows the plot of the measured elevation (in degrees) of the tracked vehicles by the proposed system. We can see a similar trend in the measurements, but this time the estimated measurements are much less noisy than the Radar. The experiments show that the Radar is good at the estimation of velocity and range of objects, whereas the proposed system had noisy measurements for velocity, and the noise is comparatively less for radial distance estimation. The main advantage of the proposed system over the Radar is that it can classify objects, and the object's azimuth and elevation measurements are more accurate than that of Radar's. Fusing measurements from both systems will increase the perception capability of the vehicle.


(a) Disparity map.

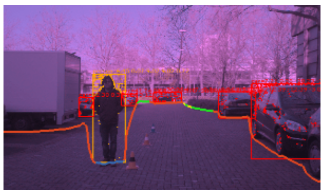
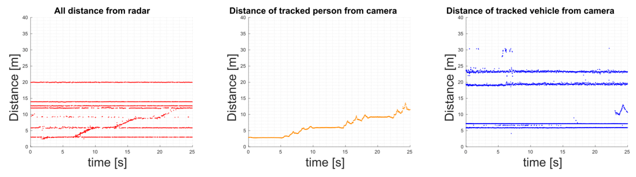

(b) Detection and tracking output.

Fig. 10: A controlled outdoor scenario, where a person moved from 3m to 6m to 9m to 12m away from the ego vehicle. Figure 10a shows the disparity map of the scene, and Figure 10b shows the output of the proposed system.

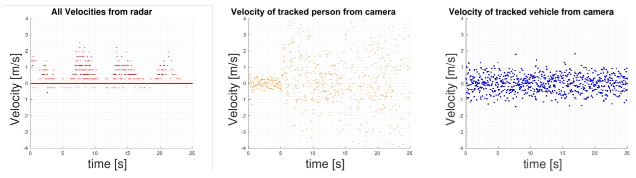
B. *Traffic Environment*

In the scenario depicted in Figure 15, the ego vehicle was following the lead vehicle on a highway. We can see that the object perception module detects the vehicles (red bounding



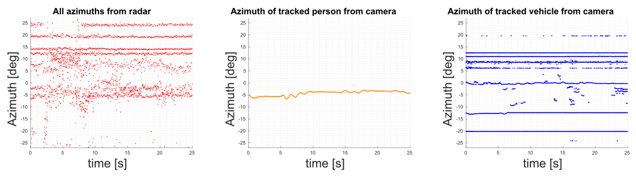
(a) (b) (c)

Fig. 11: Figure 11a shows the plot of the measured relative radial distance of objects detected by the Radar. Figures 11b and 11c show the plot of the measured relative radial distance of the tracked person and vehicles by the proposed system, respectively.



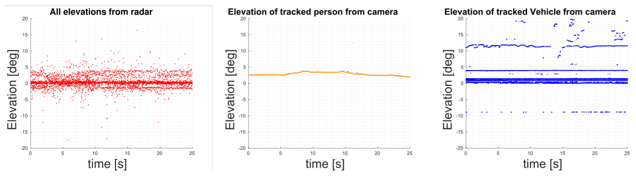
(a) (b) (c)

Fig. 12: Figure 12a shows the plot of the measured relative velocity of objects detected by the Radar. Figures 12b and 12c show the plot of the measured relative velocity of the tracked person and vehicles by the proposed system, respectively.



(a) (b) (c)

Fig. 13: Figure 13a shows the plot of the measured azimuth of objects detected by the Radar. Figures 13b and 13c show the plot of the measured azimuth of the tracked person and vehicles by the proposed system, respectively.



(a) (b) (c)

Fig. 14: Figure 14a shows the plot of the measured elevation of objects detected by the Radar. Figures 14b and 14c show the plot of the measured elevation of the tracked person and vehicles by the proposed system, respectively.

boxes), and traffic lights (magenta bounding boxes) in the scene accurately. The lane perception module detects the left ego lane (red line) and right ego lane (blue line) accurately. The free space detection module also detects the drivable free space in front of the ego vehicle and is represented as a continuous curve from the left to the right side of the image.

C. *Processing Time*

We compare the processing time of the proposed system, on the Ubuntu 16.04 (x86_64 architecture) and Drive PX 2

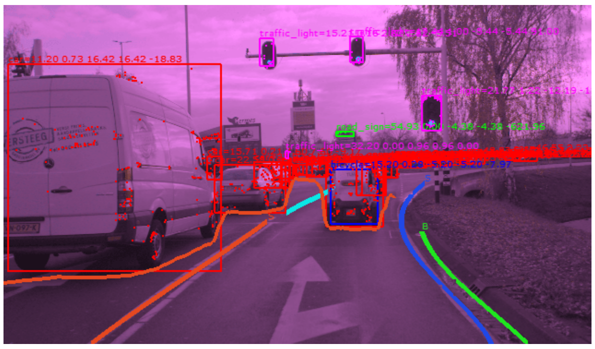

car=11.20 0.73 16.42 16.42 -18.83
traffic_light=15.2
traffic_light=21.77 1.22 -12.19 -1
road_sign=54.93 0.00 -4.38 -4.38 -621.96
traffic_light=32.20 0.00 0.96 0.96 0.00
bicycle=15.20 0.28 -5.20 -5.20 -7.97
S
S
B

Fig. 15: Results of the proposed system, in highway traffic scenario.

TABLE I: Performance comparison of frameworks in terms of processing time (in milliseconds).

Platform (architecture)	Nvidia Deep Neural Networks			Stereo Vision System
	Object Detection	Lane Detection	Free Space Detection	
Ubuntu16.04 (x86_64)	11	03	01	23
Drive PX 2 (aarch64)	34	06	04	138

(aarch64 architecture) platform with Nvidia DriveWorks 1.2, CUDA 9.2, and CuDNN 7.4.1 library, are shown in Table I. The processing time of the proposed system is 23 ms (43 Hz) on a PC with Intel Core i7 CPU (Ubuntu 16.04 LTS), Nvidia TITAN Xp GPU card with PASCAL architecture (Nvidia graphics driver 396 for x86 platform), and 138 ms (7.2 Hz) on the Drive PX 2 platform, which is suitable for various low-speed cooperative automated vehicle applications.

IV. Conclusions

In this paper, we presented the architecture design and development of an on-board stereo vision system for cooperative automated vehicles. The experimental result shows that Radar provides more accurate or less noisy range and velocity measurements than the proposed system. Whereas the proposed system's azimuth and elevation angle measurements are more accurate than that of Radar. The main advantage over the Radar system is that the proposed system can accurately classify objects and estimate radial distance, azimuth, and elevation angles. The proposed system runs at 7.4 Hz on Drive PX 2 automotive-grade platform in a real-time environment and it can be used in cooperative automated driving.

ACKNOWLEDGMENT

The authors would like to give recognition to the remainder of the i-CAVE team without whom the project would not have been possible. Moreover, we would like to thank Varun Khattar for his support in testing and analysis with the Radar system. This research work is part of the i-CAVE research programme within the Sensing, Mapping and Localization project (project number 10024085). This i-CAVE programme is funded by NWO (Netherlands Organisation for Scientific Research).

References

[1] E. Yurtsever, J. Lambert, A. Carballo, and K. Takeda, "A survey of autonomous driving: common practices and emerging technologies," *arXiv preprint arXiv:1906.05113*, 2019.
[2] Smart, Green and Integrated Transport, Horizon 2020, European Union, https://ec.europa.eu/programmes/horizon2020/en/h2020-section/smart-green-and-integrated-transport, [Online], 2020.
[3] I. Y. Noy, D. Shinar, and W. J. Horrey, "Automated driving: Safety blind spots," *Safety science*, vol. 102, pp. 68–78, 2018.
[4] J. R. Treat, N. Tumbas, S. McDonald, D. Shinar, R. D. Hume, R. Mayer, R. Stansifer, and N. Castellan, "Tri-level study of the causes of traffic accidents: final report. executive summary." Indiana University, Bloomington, Institute for Research in Public Safety, Tech. Rep., 1979.
[5] Q. Chen, S. Tang, Q. Yang, and S. Fu, "Cooper: Cooperative perception for connected autonomous vehicles based on 3d point clouds," in *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2019, pp. 514–524.
[6] Q. Zhang, Y. Wang, X. Zhang, L. Liu, X. Wu, W. Shi, and H. Zhong, "Openvdap: An open vehicular data analytics platform for cavs," in *2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS)*, July 2018, pp. 1310–1320.
[7] C. Bergenhem, S. Shladover, E. Coelingh, C. Englund, and S. Tsugawa, "Overview of platooning systems," in *Proceedings of the 19th ITS World Congress, Oct 22-26, Vienna, Austria (2012)*, 2012.
[8] Integrated Cooperative Automated Vehicles (i-CAVE) project, https://i-cave.nl/, [Online], 2019.
[9] T. van der Sande and H. Nijmeijer, "From cooperative to autonomous vehicles," in *Sensing and Control for Autonomous Vehicles*. Springer, 2017, pp. 435–452.
[10] i-Cave participation ITS European Congress 2019, https://i-cave.nl/i-cave-deelname-its-european-congress-2019/, [Online], 2019.
[11] J. Ploeg and R. de Haan, "Cooperative automated driving: from platooning to maneuvering," in *5th International Conference on Vehicle Technology and Intelligent Transport Systems, VEHITS 2019*. SCITEPRESS-Science and Technology Publications, Lda., 2019, pp. 5–10.
[12] J. Ploeg, "Analysis and design of controllers for cooperative and automated driving," 2014.
[13] N. Kemsaram, A. Das, and G. Dubbelman, "An integrated framework for autonomous driving: object detection, lane detection, and free space detection," in *2019 Third World Conference on Smart Trends in Systems Security and Sustainablity (WorldS4)*. IEEE, 2019, pp. 260–265.
[14] R. Dominguez, E. Onieva, J. Alonso, J. Villagra, and C. Gonzalez, "Lidar based perception solution for autonomous vehicles," in *2011 11th International Conference on Intelligent Systems Design and Applications*. IEEE, 2011, pp. 790–795.
[15] O. Schumann, J. Lombacher, M. Hahn, C. Wohler, and J. Dickmann, "Scene understanding with automotive radar," *IEEE Transactions on Intelligent Vehicles*, pp. 1–1, 2019.
[16] J. Fan, X. Zhu, and H. Yang, "Three-dimensional real-time object perception based on a 16-beam lidar for an autonomous driving car," in *2018 IEEE International Conference on Real-time Computing and Robotics (RCAR)*. IEEE, 2018, pp. 224–229.
[17] J. Dickmann, N. Appenrodt, H. Bloecher, C. Brenk, T. Hackbarth, M. Hahn, J. Klappstein, M. Muntzinger, and A. Sailer, "Radar contribution to highly automated driving," in *2014 44th European Microwave Conference*, Oct 2014, pp. 1715–1718.
[18] N. Kemsaram, A. Das, and G. Dubbelman, "A stereo perception framework for autonomous vehicles," in *Proceedings of IEEE 91st Vehicular Technology Conference 2020 Spring (VTC2020-Spring)*. IEEE, 2020.
[19] D. Hernandez-Juarez, A. Chacón, A. Espinosa, D. Vázquez, J. C. Moure, and A. M. López, "Embedded real-time stereo estimation via semi-global matching on the gpu," *Procedia Computer Science*, vol. 80, pp. 143–153, 2016.
[20] Nvidia DriveWorks Development Guide, https://developer.nvidia.com/driveworks-docs/, [Online], 2020.
[21] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.
[22] J. A. Fernandes and J. C. Neves, "Angle invariance for distance measurements using a single camera," in *2006 IEEE International Symposium on Industrial Electronics*, vol. 1. IEEE, 2006, pp. 676–680.