# A Real-Time Support with Haptic Feedback for Safer Driving using Monocular Camera

Giorgio De Magistris$^{1[0000-0002-3076-4509]}$, Lorenzo Guercio[1], Francesco Starna[1], Samuele Russo$^{2[0000-0002-1846-9996]}$, Natalia Kryvinska[3], and Christian Napoli$^{1[0000-0002-3336-5853]}$

[1] Department of Computer, Control, and Management Engineering, Sapienza University of Rome, via Ariosto 25, 00185 Rome, Italy
{demagistris,cnapoli}@diag.uniroma1.it

[2] Department of Psychology, Sapienza University of Rome, via dei Marsi 78, Roma 00185, Italy
samuele.russo@uniroma1.it

[3] Comenius University in Bratislava, Faculty of Management, Slovakia
Natalia.Kryvinska@uniba.sk

**Abstract.** Each year, car accidents impact billions of people, resulting in numerous casualties. Consequently, road safety remains a top priority for nations worldwide. This project aims to enhance driver safety through a feedback system that relies solely on a monocular camera mounted atop the vehicle. The proposed system is a real-time application designed to warn drivers of imminent road hazards, which are classified by their level of risk. Our method employs various computer vision techniques and incorporates a simple 2D-3D correspondence to estimate the longitudinal and lateral distances of objects ahead of the vehicle, under certain simplifying assumptions. The system conducts a comprehensive danger analysis by evaluating potential hazards within the vehicle's path. Depending on the danger level, warnings are delivered to the driver with varying degrees of invasiveness, using haptic feedback. The proposed method was tested on the KITTI dataset, yielding positive results.

**Keywords:** Haptic Feedback · Computer Vision · Deep Learning

## 1 Introduction

In recent years, Artificial Intelligence for autonomous driving systems has become more and more important and is having a huge impact in our life. The most recent driving systems have been equipped with driver assistance functions such as Lane Keeping Assistant (LKA) [18, 29, 6, 2], Adaptive Cruise Control (ACC) [30] and Brake Assist System (BAS) [17], in order to increase safety in driving. All of these assistant functions rely on Computer Vision algorithms, in particular on object detection and distance estimation [7]. Those techniques may rely on different kinds of sensors, such as a LiDAR scanner [38, 24], monocular [9] or stereo [23] cameras, and GPS [21]. The LiDAR scanner creates a 3D map of the surrounding environment and it is more accurate than 2D information, but it can be very expensive. GPS alone is not adequate for real time assistance. Camera images, instead, provide enough data to build a driver assistance function, and it is also a cheap sensor that can be easily installed in every vehicle. In this paper we focus on developing a haptic feedback function that is complementary with the Emergency Brake Assistant (EBA), to ensure safe driving and avoid potential risks before the latter is activated. At each frame we first take the image coming from a monocular camera, placed above the vehicle roof. Then we detect and classify the objects in the scene using YOLOv4, that is a one-stage object detector. We manipulate the camera calibration matrix in order to recover longitudinal and lateral distances from the vehicle to the objects detected. After that, we apply an object tracking algorithm on the objects in order to track their motion. Then, for each object detected, we evaluate the potential danger based on different specific criteria. Finally, we warn the driver with a haptic feedback that is proportional to the level of danger. In particular, the higher the value the more intrusive the feedback, until we reach the maximum danger, where the EBA function will break the vehicle to avoid a hazard. We evaluate our method on the KITTI raw dataset, using mean average precision (mAP) for object detection, and root mean square error (RMSE) for distance estimation.

## 2 Related Works

Recent advancements in haptic feedback systems have improved driving safety by providing non-visual, non-auditory alerts. These systems use tactile or kinesthetic feedback via the steering wheel, seat, or pedals to communicate critical information like lane departures, proximity warnings, and collision risks. Haptic feedback keeps drivers visually focused on the road while delivering essential cues, proving more effective than visual or auditory alerts in some high-load situations. As noted in [11], haptic systems are classified as either assistance systems, which provide continuous feedback for tasks like navigation or parking, or warning systems, which alert drivers to immediate dangers like collisions. These systems reduce driver response times and enhance spatial awareness, making them essential in modern Advanced Driver Assistance Systems (ADAS). Studies show that combining haptic feedback with other sensory modalities further enhances performance and reaction times in complex environments.

### 2.1 Object Detectors

State-of-the-art object detectors are mainly divided into two-stage and one-stage detectors. Two-stage detectors (i) use a Region Proposal Network (RPN) to identify regions of interest, and (ii) classify objects and refine bounding boxes. These methods, such as R-CNN [13] and Faster R-CNN [35], are accurate but slower [4, 5]. In contrast, one-stage detectors treat detection as a regression problem, predicting bounding boxes and class probabilities in a single step, making them faster but generally less accurate [36]. Notable one-stage detectors include SSD [27, 8] and YOLO [33]. For real-time systems, achieving over 10 FPS is crucial, as the human visual system perceives individual images below this threshold and motion above it [31]. Given its speed and performance, YOLOv4 [3] was chosen for our system.

For further details, refer to the survey by [19], which covers various object detection methods.

### 2.2 Distance Estimation

Current distance estimation in autonomous driving often relies on LiDAR, which, like radar, calculates distance via time of flight. However, LiDAR is expensive. In our work, we focus on monocular cameras, a cost-effective alternative. Stereo cameras usually estimate depth through triangulation, but monocular cameras can estimate 3D distances using techniques like inverse perspective mapping (IPM) and 2D-3D correspondences under simplifying assumptions. This monocular approach, while less complex than stereo setups, provides adequate distance estimation for most road scenarios. Stereo cameras require more complex hardware and calibration to maintain alignment, adding to setup and maintenance complexity. Monocular systems avoid these challenges.

Many recent works use IPM to compute longitudinal distances. For example, [20] combines IPM and YOLO for object detection and distance estimation. Similarly, [32] uses IPM, camera matrices, and lane detection to compute Euclidean distances, while [1] employs IPM and HSV colormap to define the region of interest and retrieve distances. However, IPM depends on the road's vanishing point, which can fail in curves.

Machine learning offers another approach to distance estimation. DisNet [15] uses YOLO to train a neural network for supervised distance estimation, providing a dataset with 2D bounding boxes and distances. [22] developed FisheyeDistanceNet, which estimates depth from fisheye images. While effective, machine learning methods require extensive training and data.

Our approach is faster and simpler. By manipulating the camera matrix and using 3D-2D correspondences from the Pinhole camera model, we efficiently estimate longitudinal and lateral distances.

## 3 Pipeline

In this section we present our work. Each step of the pipeline (illustrated in Figure 1) is designed to be executed frame by frame in real time during the driving.

### 3.1 Object Detection

To evaluate potential dangers, we first need to detect all objects in the scene. Our approach focuses on the one-stage detector YOLOv4, the fourth improved

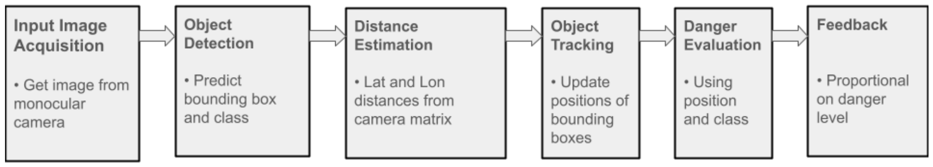
Input Image Acquisition
• Get image from monocular camera
Object Detection
• Predict bounding box and class
Distance Estimation
• Lat and Lon distances from camera matrix
Object Tracking
• Update positions of bounding boxes
Danger Evaluation
• Using position and class
Feedback
• Proportional on danger level

Fig. 1: The complete pipeline of the proposed method.

version of YOLO, chosen based on its balance between inference time and average precision. A modern one-stage object detector generally consists of three key components. First, the backbone, which serves as a feature extractor, is typically pre-trained on large datasets such as ImageNet [10]. Second, the neck is responsible for analyzing and refining features from different stages of the backbone. Finally, the head is tasked with generating bounding boxes and making class predictions.

In the specific case of YOLOv4, the backbone used is CSPDarknet53, an efficient architecture designed for feature extraction. The neck integrates Spatial Pyramid Pooling (SPP) and PANet, which aggregate and process features across multiple scales to enhance detection accuracy. The head is derived from YOLOv3, responsible for generating precise bounding boxes and classifying objects within them. This combination allows YOLOv4 to provide a strong balance of accuracy and speed, making it ideal for real-time object detection.

**Backbone** The Cross Stage Partial Network (CSPNet) [37] was introduced to reduce the computation of heavy neural networks, which is fundamental to develop real-time applications on small devices. YOLOv4 applied CSPNet to Darknet53, which is a convolutional neural network using residual connections introduced in YOLOv3 [34].

**Neck** As an additional block placed after the backbone YOLOv4 implements Spatial Pyramid Pooling (SPP) [16], which is a more robust method to image deformations (crop/warp) for both object detection and classification. To complete the section, the Path Aggregation Network (PANet) [26] is added to enhance the entire feature hierarchy, in order to let useful information in each feature level propagate directly to the following proposal subnetworks.

**Head** YOLO divides the image into an S $\times$ S grid and for each grid cell predicts B bounding boxes, each one consisting of 5 predictions: $x$, $y$, $w$, $h$ and *confidence*. Each grid cell also predicts C conditional class probabilities: *Prob (Class*$|$*Object)*. YOLOv3's main idea is totally based on the original work, even if it predicts boxes across three different scales, using a similar concept to Feature Pyramid Networks [25]. For this reason, the predictions for the third scale benefit from all the prior computation as well as fine grained features from early on in the network. The 2D bounding box predictions are sufficient to estimate the distances from the vehicle to all the objects in the scene, since we need to know only the lower side of the box, which represents the contact point of the object with the road.

## 3.2 Distance Estimation

Before going into details, we have to state two simplifying assumptions: (a1) the road on which the vehicle and all the objects in the scene lie, must be a planar surface, (a2) the camera installed on the vehicle must be stationary. These simplifications allowed us to develop an efficient, easy and fast computational method for distance estimation.

**Camera Matrix** *Multiple View Geometry in Computer Vision* [14] describes how the pinhole camera model maps world points to image points. Using homogeneous coordinates we can write:

$$P = K[R \mid t] = \begin{bmatrix} f_x & s & x_0 \\ 0 & f_y & y_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} R_{11} & R_{12} & R_{13} & t_1 \\ R_{21} & R_{22} & R_{23} & t_2 \\ R_{31} & R_{32} & R_{33} & t_3 \end{bmatrix}$$

where: K is the 3x3 camera calibration matrix containing the intrinsic parameters, describing the focal length, the optical center, and the skew coefficient, and R and t are the extrinsic parameters, namely rotation and translation of a rigid transformation from 3D world coordinate system to the 3D camera's coordinate system.

In order to obtain the camera matrix, it is necessary to perform a camera calibration process that can be done in different ways and it is implemented in computer vision libraries like OpenCV. The camera matrix P defines how a world point $X$ is mapped to an image point $x$:

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = P \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \qquad (1)$$

## 3.3 2D-3D Correspondence

We start by defining the camera frame RF$_c$, which we will use from now on for every transformation, it is a right-hand coordinate system with the y axis pointing down (x axis points to the left and the z axis points out of the screen). Given the camera matrix $P$, the height of the camera from the road to the vehicle roof $h$, and the longitudinal distance from the camera to the front bumper of the car $b$, we can now manipulate the camera in order to obtain a one-to-one correspondence that maps image points $x$ to world points on the road $X$. We first translate the camera by multiplication with a transformation matrix $T$:

$$P_t = PT = P \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & h \\ 0 & 0 & 1 & b \end{bmatrix}$$

where $b = \mathit{camera2bumper}$ and $h = \mathit{camera\ height}$

As we can see in Image 2, we have translated the camera matrix on the road (height = 0) and towards the front side of the vehicle, through sequence of homogeneous transformations, so that the distances are computed directly from the central point of the bumper. At this point the assumptions come handy. Thanks to (a1) we simply eliminate from the camera matrix $\mathrm{P_t}$ the $Y$ column, meaning that all the points projected to the real world have height equal to zero. This is beneficial for computing the inverse projective mapping of (1), from world point to image point:

$$\begin{pmatrix} X \\ Z \\ 1 \end{pmatrix} = P_{t,y=0}^{-1} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \qquad (2)$$

Thanks to (a2) we are able to define this mapping in every camera frame, ignoring all the disturbances due to road irregularities and vehicle movements. With this mapping it is easy to estimate longitudinal and lateral distances. Given a generic 2D bounding box (*x1*, *x2*, *x3*, *x4*), coming from the object detection step, we take the two contact points of the object with the road in image coordinates, and we apply the inverse mapping (2), which gives us the left end and right end sides in world coordinates. At this point we take the midpoint between the two ($t_{long}$, $t_{lat}$, 1), which corresponds to longitudinal and lateral planar distances from the vehicle bumper to that object.

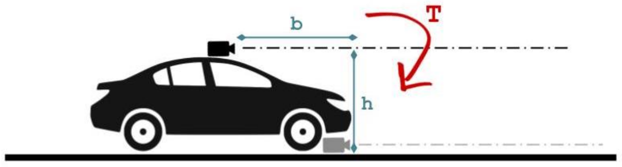

Fig. 2: Visualization of the T matrix transformation

## 3.4 Object Tracking

With the purpose of obtaining more specific information about the objects in the scene, we implemented an Euclidean object tracking system. It works by storing a dictionary of objects' longitudinal and lateral distances, with their IDs as keys.

For each object detected in a generic frame, the tracker is updated, taking as input the vector ($t_{long}$, $t_{lat}$) of that object. The tracker compares the vector with the stored dictionary, using Euclidean difference, and if the new vector is sufficiently "close" to something, the dictionary entry of the corresponding point is updated with the new distances, otherwise it is marked as a new ID. The tracker works really well with objects detected within a certain lateral distance range, beyond which we are no more interested in tracking. Once we have the IDs of the objects in the scene, we can compute a few more properties such as the relative longitudinal and lateral velocities ($v_{long}$, $v_{lat}$) of the objects with respect to the vehicle, which are important in the danger evaluation phase for making predictions of potential collisions.

## 3.5 Danger Evaluation

The evaluation of potential danger situations, in order to provide a danger haptic feedback (DHF) complementary to the EBA function, is the core of our project. We apply different criteria according to the following lateral distance subdivisions (distances are considered laterally in both directions):1) **danger zone**: from 0 to 2 *m*. These are all the objects detected right in front of the vehicle, considering a total span of 4 meters; 2) **attention zone**: from 2 to 5 *m*. These are all the objects detected just close to the vehicle; 3) **safe zone**: from 5 to 10 *m*. Further objects.

The zone limits have been chosen according to some considerations: 1) the danger zone corresponds to the maximum lane width (3.75 *m*) approximated to the next integer, 2) the attention zone comprises the next lane and an eventual sidewalk, 3) the safe zone extends till the predictions of the object recognition system give reliable results laterally (10 *m*). Distances are taken in modulus and computed starting from the midpoint of the front bumper of the vehicle, positive to the right.

Table 1: Table of vulnerability coefficient for evaluating dangerousness for different classes.

| | Car | Van | Truck | Tram | Misc | Pedestrian | Cyclist |
|---|---|---|---|---|---|---|---|
| **Vulnerability** | 1.0 | 1.0 | 0.8 | 0.9 | 1.0 | 1.5 | 1.5 |

Table 2: Different level of haptic feedbacks depending on the dangerousness value

| **Dangerousness** $D$ | $7 \leq D < 8$ | $8 \leq D < 9$ | $9 \leq D < 10$ | $\geq 10$ |
|---|---|---|---|---|
| **Haptic Feedback** | 1-level | 2-level | 3-level | Break (EBA) |

We assigned to each zone a support coefficient and to each object class a vulnerability coefficient. Such coefficients are: 1 for the *Danger Zone*, 0.8 for the *Attention Zone* and 0.5 for the *Safe Zone*. The support coefficients add a reduction contribution to dangerousness, according to the zone where the object is detected. The further the zone, the less dangerous the situation. The vulnerability coefficients take into consideration the class of the detected object. Colliding a cyclist or a pedestrian, for example, is more dangerous than colliding a truck, because it may cause a serious damage to the integrity and security of people with a higher probability.

**Evaluation** For each detected object, we evaluate the dangerousness of a potential accident in the following manner:

$$\boldsymbol{D} = remap(V_{cr} * \alpha_v * \beta_z)$$

where D is the evaluated dangerousnees, $V_{cr}$ is the criterion value, $\alpha_v$ is the vulnerability class coefficient, $\beta_z$ is the zone coefficient, and the *remap* function remaps the value to the range 0, 10. Essentially the output value of the chosen criteria is smoothed by the two coefficients and finally remapped to a valuable range. The criterion are: stopping (longitudinal) distance, Euclidean distance, intersection distance.

In this way we restrict the value of the multiplication from 0 to 10 applying different criteria according to the zone on which the object is laying. In every zone, we make use of the longitudinal velocity of the vehicle $v_e$, which can be retrieved by sensors, such as Active Sensor Bearing (ASB), GPS or Inertial Measurement Unit (IMU).

**Danger zone**. When considering an object detected in front of the vehicle, the danger comes from the possible collision due to insufficient *stopping distance* $t_{stop}$. In fact, in each frame, we take into consideration the closest (longitudinally) objects detected in the scene, taking their longitudinal distance. The smaller $t_{stop}$ - $t_{long}$, the higher the probabilities to collide, if the object in front of the vehicle unexpectedly stops. We compute the stopping distance as the sum of the perception-reaction distance and the braking distance as:

$$v_e * t_{reaction} + v_e^2/(2 * \mu * g)$$

where $t_{reaction}$ is the reaction time, $\mu$ is the friction coefficient and $g$ is the gravity of the earth. Reasonable values for $t_{reaction}$ and $\mu$ are respectively 1 second and 0.8, but these can vary according to the age of the vehicle driver and to what kind of vehicle he is driving.

**Attention zone**. In the nearby zone, the danger comes from the possibility of some objects to unexpectedly cross the danger zone and appear in front of the vehicle. In this case we increase or decrease the danger according to the following rules:

- If some object is going towards the danger zone and if the predicted lateral shift of that object intersects the predicted longitudinal shift of the vehicle (the directions of the object and the vehicle intersect), and it happens in less than 3 seconds, the danger increase inversely proportional to the intersection time. We call this criteria *intersection distance*.
- Otherwise, we consider only the *Euclidean distance*, from the vehicle to the objects laying on the attention zone.

**Safe zone**. Objects detected at a lateral distance higher than 5 meters are not dangerous at all. In order to preserve the previous criteria and provide continuity to danger evaluation, especially in the case of objects crossing from the safe to the attention zone, we use as a criterion the *Euclidean distance*, from the vehicle to the objects laying on the safe zone. Image 3 shows some examples of danger evaluation, in different scenarios.

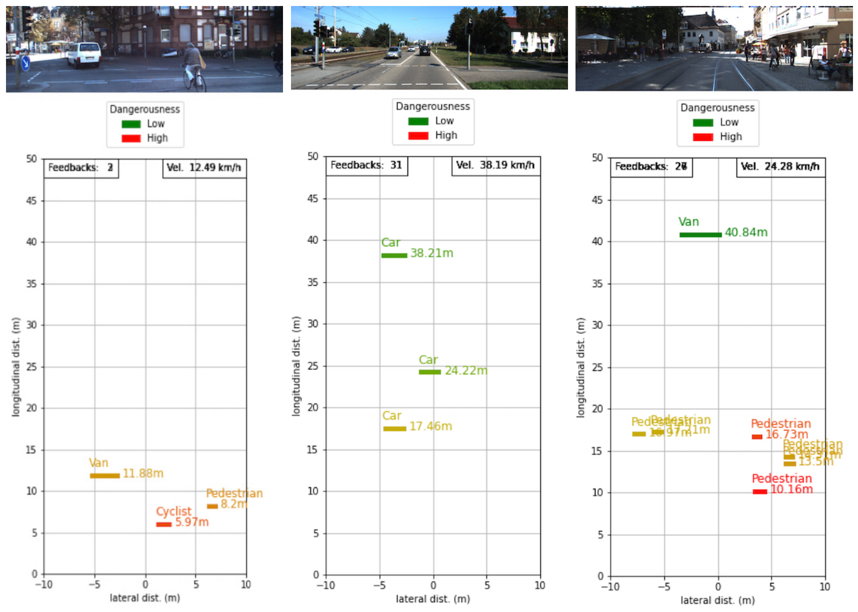

Fig. 3: Examples of Danger Evaluation

### 3.6 Haptic Feedback

The value of dangerousness computed in the last step is used to give feedback to the driver. Different kinds of actuators for our feedback system are possible, e.g.

vibration system installed on the steering wheel, smart band on the wrist, etc. Every system is capable of generating a vibration on a intensity scale from 1 to 3. The values of dangerousness may be relevant and worthy of attention when greater than 7. According to that value, we can map our haptic feedback system together with the EBA as shown in Table 2.

## 4 Experiments

In this section we report the experiments and the results that we obtained, also mentioning the data used and the hardware we used.

### 4.1 Dataset

We tested our work on the KITTI [dataset][12], which provides an annotated dataset for 2D and 3D object detection, and also a raw dataset for testing purposes. We used the object data for training and validating the object detection model, while the raw data was used for testing the object detection, the distance estimation and the object tracking, since the IMU information is provided only for the least. We focused on the scenarios 0005, 0015, and 0091 of the raw data, which respectively represent urban, highway and limited traffic zone scenarios.

### 4.2 Performances

We developed and tested our project on a Tesla T4 GPU with 12 GB of memory. With this architecture our pipeline runs at 15 FPS.

**Object Detection** We trained the YOLOv4 model on KITTI 2D object data, consisting of 7481 images randomly divided into 80% train, 10% validation and 10% test. The network has been initialized with pre-trained weights on ImageNet. Training has been done using stochastic gradient descent with warm restart (SGDR) [28] optimizer, with momentum 0.9, learning rate $10^{-3}$, and decay $5*10^{-4}$, with batch size 64, for a total of 7500 iterations (about 60 epochs). We evaluated the object detection on a small amount of data (about 750 images), because unfortunately the test set is not annotated. We obtained a mean Average Precision (mAP) of 92.8%.

**Distance Estimation** We evaluated the distances with the RMSE metric, distinguishing between short/long and longitudinal/lateral distances. Longitudinal distances are evaluated within 50 meters, while lateral distances within the zone dimensions. Table 3 shows the results of the RMSE metric over two different KITTI videos, while in Image 4 the error plots can be seen. Image 5 shows some qualitative results of detection and distance estimation.

Table 3: Performances on two different scenarios. The 0005 is calculated considering distances from a front cyclist. The 0015 is calculated considering distances from a front car. Empty values are reported when measures were not available or not enough to compute the relative metric.

| **Scenario** | **Long. Distance** | **Long. RMSE** | **Lateral. Distance** | **Lateral RMSE** |
|---|---|---|---|---|
| | <10m | 0.908 | <2m | 0.228 |
| | 10m-30m | 1.931 | 2m-5m | 0.306 |
| 0005 | 30m-50m | - | 5m-10m | - |
| | Total | 1.492 | Total | 0.262 |
| | <10m | - | <2m | 0.151 |
| | 10m-30m | 2.425 | 2m-5m | - |
| 0015 | 30m-50m | 3.785 | 5m-10m | - |
| | Total | 3.025 | Total | 0.151 |

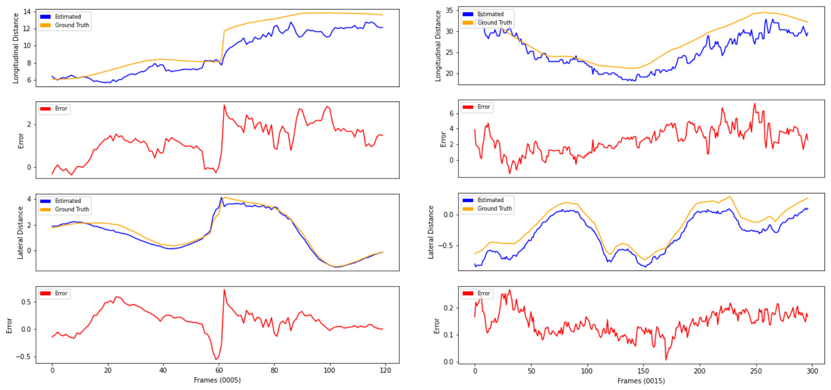

Fig. 4: distance performances in two different scenarios

## 5 Conclusions

The proposed method has proved considerably effective in the conditions established at the beginning of the paper and the budget necessary for its implementation is very low. However, some aspects must be considered before applying the method in a real scenario: for example the method totally rely on object detection: if there are errors in this step of the pipeline, there is no other type of check is performed. Moreover the object detector works up to more or less 50 meters: it would be better if the detection reached about 100 meters (in order to be more effective in highways for example). The proposed method may not work properly in case of adverse visibility conditions or at night, due to the worse performances of the object detection. Moreover, due to the initial assumptions, we have a considerable margin of error when the machine "goes up and down" due to the bumps.

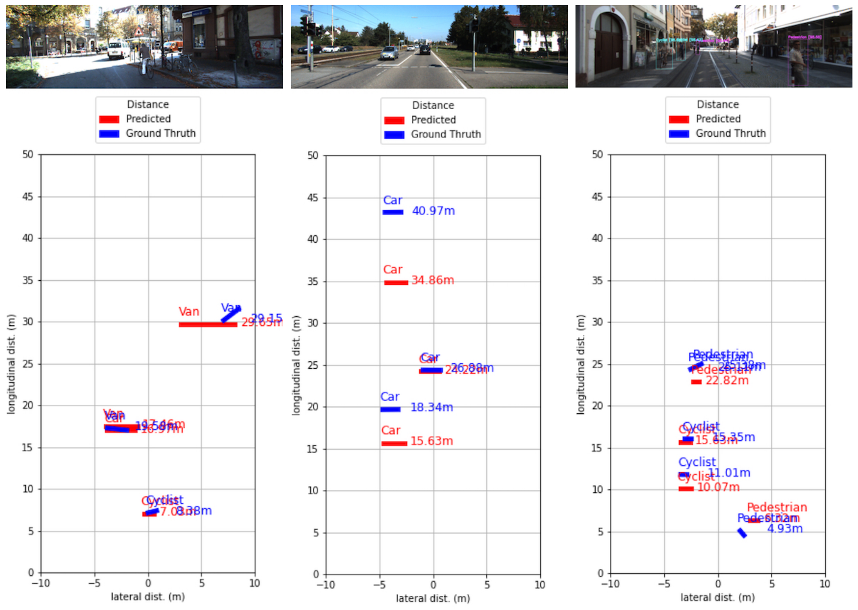

Fig. 5: Detection and distance estimation plots

## References

1. Adamshuk, R., Carvalho, D., Neme, J.H.Z., Margraf, E., Okida, S., Tusset, A., Santos, M.M., Amaral, R., Ventura, A., Carvalho, S.: On the applicability of inverse perspective mapping for the forward distance estimation based on the hsv colormap. In: 2017 IEEE International Conference on Industrial Technology (ICIT). pp. 1036–1041 (2017). https://doi.org/10.1109/ICIT.2017.7915504
2. Alfarano, A., De Magistris, G., Mongelli, L., Russo, S., Starczewski, J., Napoli, C.: A novel convmixer transformer based architecture for violent behavior detection. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) **14126 LNAI**, 3 – 16 (2023). https://doi.org/10.1007/978-3-031-42508-0_1
3. Bochkovskiy, A., Wang, C.Y., Liao, H.Y.M.: Yolov4: Optimal speed and accuracy of object detection (2020)
4. Bonanno, F., Capizzi, G., Coco, S., Napoli, C., Laudani, A., Sciuto, G.L.: Optimal thicknesses determination in a multilayer structure to improve the spp efficiency for photovoltaic devices by an hybrid fem - cascade neural network based approach. In: 2014 International Symposium on Power Electronics, Electrical Drives, Automation and Motion, SPEEDAM 2014. pp. 355 – 362 (2014). https://doi.org/10.1109/SPEEDAM.2014.6872103
5. Bonanno, F., Capizzi, G., Sciuto, G.L., Napoli, C., Pappalardo, G., Tramontana, E.: A cascade neural network architecture investigating surface plasmon polaritons propagation for thin metals in openmp. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioin-

formatics) **8467 LNAI**(PART 1), 22 – 33 (2014). https://doi.org/10.1007/978-3-319-07173-2_3
6. Brandizzi, N., Russo, S., Brociek, R., Wajda, A.: First studies to apply the theory of mind theory to green and smart mobility by using gaussian area clustering. In: CEUR Workshop Proceedings. vol. 3118, pp. 71 – 76 (2021)
7. Brociek, R., Magistris, G.D., Cardia, F., Coppa, F., Russo, S.: Contagion prevention of covid-19 by means of touch detection for retail stores. In: CEUR Workshop Proceedings. vol. 3092, pp. 89 – 94 (2021)
8. Capizzi, G., Bonanno, F., Napoli, C.: A wavelet based prediction of wind and solar energy for long-term simulation of integrated generation systems. In: SPEEDAM 2010 - International Symposium on Power Electronics, Electrical Drives, Automation and Motion. pp. 586 – 592 (2010). https://doi.org/10.1109/SPEEDAM.2010.5542259
9. Chen, X., Kundu, K., Zhang, Z., Ma, H., Fidler, S., Urtasun, R.: Monocular 3d object detection for autonomous driving. In: 2016 IEEE Conference on Computer Vision and Patter Recognition (CVPR). pp. 2147–2156 (2016). https://doi.org/10.1109/CVPR.2016.236
10. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition. pp. 248–255 (2009)
11. Gaffary, Y., Lécuyer, A.: The use of haptic and tactile information in the car to improve driving safety: A review of current technologies. Frontiers in ICT **5**, 5 (2018)
12. Geiger, A., Lenz, P., Stiller, C., Urtasun, R.: Vision meets robotics: The kitti dataset. International Journal of Robotics Research (IJRR) (2013), http://www.cvlibs.net/datasets/kitti/index.php
13. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation (2014)
14. Hartley, R., Zisserman, A.: Multiple View Geometry in Computer Vision. Cambridge University Press, 2 edn. (2004). https://doi.org/10.1017/CBO9780511811685
15. Haseeb, M.A., Guan, J., Ristic-Durrant, D., Gräser, A.: Disnet: A novel method for distance estimation from monocular camera (2018)
16. He, K., Zhang, X., Ren, S., Sun, J.: Spatial pyramid pooling in deep convolutional networks for visual recognition. Lecture Notes in Computer Science p. 346–361 (2014). https://doi.org/10.1007/978-3-319-10578-9_23
17. Hirose, T., Taniguchi, T., Hatano, T., Takahashi, K., Tanaka, N.: A study on the effect of brake assist systems (bas). SAE International Journal of Passenger Cars - Mechanical Systems **1**(1), 729–735 (apr 2008). https://doi.org/https://doi.org/10.4271/2008-01-0824, https://doi.org/10.4271/2008-01-0824
18. Ishida, S., Gayko, J.: Development, evaluation and introduction of a lane keeping assistance system. In: IEEE Intelligent Vehicles Symposium, 2004. pp. 943–944 (2004). https://doi.org/10.1109/IVS.2004.1336512
19. Jiao, L., Zhang, F., Liu, F., Yang, S., Li, L., Feng, Z., Qu, R.: A survey of deep learning-based object detection. IEEE Access **7**, 128837–128868 (2019). https://doi.org/10.1109/access.2019.2939201, http://dx.doi.org/10.1109/ACCESS.2019.2939201
20. Kim, Y., Kum, D.: Deep learning based vehicle position and orientation estimation via inverse perspective mapping image. In: 2019 IEEE Intelligent Vehicles Symposium (IV). pp. 317–323 (2019). https://doi.org/10.1109/IVS.2019.8814050

21. Kumar, S., Moore, K.B.: The evolution of global positioning system (gps) technology. Journal of Science Education and Technology **11**(1), 59–80 (Mar 2002). https://doi.org/10.1023/A:1013999415003, https://doi.org/10.1023/A:1013999415003
22. Kumar, V.R., Hiremath, S.A., Bach, M., Milz, S., Witt, C., Pinard, C., Yogamani, S., Mäder, P.: Fisheyedistancenet: Self-supervised scale-aware distance estimation using monocular fisheye camera for autonomous driving. In: 2020 IEEE International Conference on Robotics and Automation (ICRA). pp. 574–581 (2020)
23. Li, P., Chen, X., Shen, S.: Stereo r-cnn based 3d object detection for autonomous driving (2019)
24. Li, Y., Ibanez-Guzman, J.: Lidar for autonomous driving: The principles, challenges, and trends for automotive lidar and perception systems. IEEE Signal Processing Magazine **37**(4), 50–61 (2020). https://doi.org/10.1109/MSP.2020.2973615
25. Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection (2017)
26. Liu, S., Qi, L., Qin, H., Shi, J., Jia, J.: Path aggregation network for instance segmentation (2018)
27. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: Ssd: Single shot multibox detector. Lecture Notes in Computer Science p. 21–37 (2016). https://doi.org/10.1007/978-3-319-46448-0_2
28. Loshchilov, I., Hutter, F.: Sgdr: Stochastic gradient descent with warm restarts (2017)
29. Mammeri, A., Lu, G., Boukerche, A.: Design of lane keeping assist system for autonomous vehicles. In: 2015 7th International Conference on New Technologies, Mobility and Security (NTMS). pp. 1–5 (2015). https://doi.org/10.1109/NTMS.2015.7266483
30. Marsden, G., McDonald, M., Brackstone, M.: Towards an understanding of adaptive cruise control. Transportation Research Part C: Emerging Technologies **9**(1), 33–51 (2001). https://doi.org/https://doi.org/10.1016/S0968-090X(00)00022-X, https://www.sciencedirect.com/science/article/pii/S0968090X0000022X
31. Połap, D., Woźniak, M., Napoli, C., Tramontana, E.: Real-time cloud-based game management system via cuckoo search algorithm. International Journal of Electronics and Telecommunications **61**(4), 333 – 338 (2015). https://doi.org/10.1515/eletel-2015-0043
32. Qiao, D., Zulkernine, F.: Vision-based vehicle detection and distance estimation. In: 2020 IEEE Symposium Series on Computational Intelligence (SSCI). pp. 2836–2842 (2020). https://doi.org/10.1109/SSCI47803.2020.9308364
33. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection (2016)
34. Redmon, J., Farhadi, A.: Yolov3: An incremental improvement (2018)
35. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks (2016)
36. Soviany, P., Ionescu, R.T.: Optimizing the trade-off between single-stage and two-stage object detectors using image difficulty prediction (2018)
37. Wang, C.Y., Liao, H.Y.M., Yeh, I.H., Wu, Y.H., Chen, P.Y., Hsieh, J.W.: Cspnet: A new backbone that can enhance learning capability of cnn (2019)
38. Wang, Y., Chao, W., Garg, D., Hariharan, B., Campbell, M., Weinberger, K.Q.: Pseudo-lidar from visual depth estimation: Bridging the gap in 3d object detection for autonomous driving. CoRR **abs/1812.07179** (2018), http://arxiv.org/abs/1812.07179