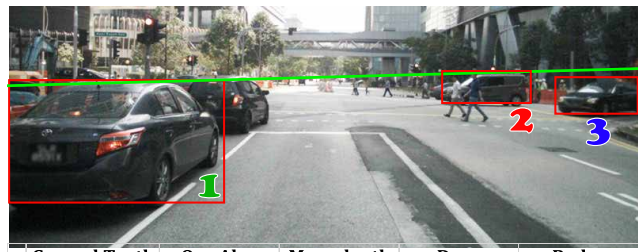
# Real-time vehicle distance estimation using single view geometry

Ahmed Ali, Ali Hassan, Afsheen Rafaqat Ali, Hussam Ullah Khan, Wajahat Kazmi, Aamer Zaheer
KeepTruckin Inc.
Lahore, Pakistan
{Ahmed.Ali, Ali.Hassan, Afsheen, Hussam.Khan, Wajahat.Kazmi, AZ} @KeepTruckin.com

## Abstract

*Distance estimation is required for advanced driver assistance systems (ADAS) as well as self-driving cars. It is crucial for obstacle avoidance, tailgating detection and accident prevention. Currently, radars and lidars are primarily used for this purpose which are either expensive or offer poor resolution. Deep learning based depth or distance estimation techniques require huge amount of data and ensuring domain invariance is a challenge. Therefore, in this paper, we propose a single view geometric approach which is lightweight and uses geometric features of the road lane markings for distance estimation that integrates well with the lane and vehicle detection modules of an existing ADAS. Our system introduces novelty on two fronts: (1) it uses cross-ratios of lane boundaries to estimate horizon (2) it determines an Inverse Perspective Mapping (IPM) and camera height from a known lane width and the detected horizon. Distances of the vehicles on the road are then calculated by back projecting image point to a ray intersecting the reconstructed road plane. For evaluation, we used lidar data as ground truth and compare the performance of our algorithm with radar as well as the state-of-the-art deep learning based monocular depth prediction algorithms. The results on three public datasets (Kitti, nuScenes and Lyft level 5) showed that the proposed system maintains a consistent RMSE between 6.10 to 7.31. It outperforms other algorithms on two of the datasets while on KITTI it falls behind one (supervised) deep learning method. Furthermore, it is computationally inexpensive and is data-domain invariant.*




| | Ground Truth | Our Algo | Monodepth | Dorn | Radar |
|---|---|---|---|---|---|
| 1 | 5.02 | 5.77 | 5.29 | 8.54 | 6.00 |
| 2 | 26.57 | 26.08 | 13.30 | 18.90 | 28.00 |
| 3 | 21.00 | 18.68 | 13.47 | 18.03 | 21.00 |

Figure 1. Sample image from nuScenes [3]. Estimated horizon (green line) and computed distances (meters) by all methods for each vehicle (1, 2 and 3) from the ego vehicle are displayed. Our method is compared against deep learning based depth estimation as well as radar. GT is the ground truth which is provided by the high resolution lidar.

## 1. Introduction

Human error is one of the major causes of driving accidents. At the heart of safe driving practices lies the driver's judgment to estimate their vehicle's braking distance w.r.t other road users. For this reason, Advanced Driver Assistance Systems (ADAS) have evolved substantially over the last few years and features such as pedestrian and tailgating detection have been introduced to alert drivers in the run-up to perilous maneuvers. Real-time logging of this information is crucial in this regard and can help fleet managers in scoring a driver's behavior. It also makes an integral part of autonomous driving in path planning, obstacle and collision avoidance and hence in overall safety assessment of the self-driving cars.

Active sensing in the form of Radars (Radio Detection and Ranging) and Lidars (Light Detection and Ranging) are a common choice for measuring distances to surrounding obstacles. Radars have ranges up to $150m$ [29] but they generally have low resolution. Lidars, on the other hand, offer higher spatial resolution [19] but they are too expensive (cheapest costs US$ $7,500$ [1]). The biggest advantage of using active sensing is its real-time operation as minimal post processing is required. Scene depth is estimated in a quick scan and simple thresholding may do the job. While improvement in the sensor technology and mass production is expected to reduce the cost of lidars by 2022 [19], they still have limitations of compromised performance in adverse weather conditions such as rain, snow and fog. In contrast, standard dashcams provide low cost and high reso-

978-1-7281-6553-0/20/$31.00 ©2020 IEEE

Authorized licensed use limited to: San Jose State University. Downloaded on February 05,2025 at 16:28:43 UTC from IEEE Xplore. Restrictions apply.

Samsara EX1008
U.S. Patent No. 11,875,580

lution RGB images of the scene but will demand intelligent post processing using robust algorithms and higher compute power on board.

In computer vision, the problem of distance estimation falls in the broader area of scene depth estimation. To this end, various data-driven and geometric approaches have been explored in literature. In data driven approaches, depth of objects is learnt through either training a supervised depth regressor [10] or unsupervised disparity estimator [13, 45]. On the other hand, geometric approaches take into account the geometry of the features in the scene such as Multi-View Stereo (MVS) and Single View Metrology [43]. MVS has remained a primary focus for depth recovery in the past [21]. It provides depth for each pixel via dense matching of more than one image. MVS is limited by the availability of texture in the scene, the efficiency of the algorithms as well as the complexity of multi-camera configuration. The accuracy of the depth map may not be consistent throughout. Single view metrology, instead, uses a single image to compute 3D measurements using perspective geometry [7]. Being ill-posed, it has remained limited to images of structured environments where a combination of parallel and orthogonal lines and surfaces are substantially available. In this paper, we use monocular dashcam images to estimate relative distances of vehicles on the road. For evaluation, we used lidar data as ground truth and established a comparison of the data-driven approaches with the radar data. We propose an efficient and data independent algorithm for real-time distance estimation for flat road surfaces using single view geometry. With the help of any available lane and vehicle detection modules (which are integral components of ADAS and self driving cars), it fully exploits the structure in the scene. We will show that it competes well with deep learning methods and under the geometric constraints, it outperforms end-to-end deep learning based methods as well as the radar data both in accuracy and efficiency making it suitable for edge devices. Since the autonomous driving is currently limited to high-way driving scenarios, it is well suited for such tasks. And except for the camera initialization phase, which is a one-time computation, it comes at almost no computational expense.

The breakdown of the paper is as follows: Section 2 provides a quick review of depth estimation literature. Section 3 discusses our approach in detail with ADAS modules for lane and vehicle detection in section 3.1, camera initialization in section 3.2 and distance estimation in section 3.3. Section 4 outlines the datasets, training and testing. Results and a discussion on the results are presented in section 4.1 and section 5 concludes the paper.

## 2. Related work

In literature, only a few works have employed approaches similar to ours. Park and Hwang [32] used monocular imaging and average vehicle width in real world to estimate horizon. Following that, the distance of the bottom of the vehicle's detected bounding box from the horizon provided a hint of its relative distance. But they assumed an already known camera height which can be difficult if done on a multitude of vehicle across a fleet. In 2017, top performing submission to TuSimple's velocity estimation challenge employed deep learning based depth prediction techniques combined with object (vehicle) detection to find relative velocities [37]. Keeping this in view, we will have to look into the research work done in the two domains i.e. our proposal of single view geometry for depth estimation and the state-of-the-art deep learning based depth prediction.

### 2.1. Single view geometry based depth estimation

In computer vision, single view based recovery of scene geometry have been thoroughly studied [4, 36, 30, 17]. Leibotwzki used geometric properties such as parallelism and orthogonality of structures to recover 3D models from one or two views without explicitly knowing the metric properties of objects [26]. Lee *et al*. showed that even in the presence of clutter, full 3D model can be recovered from a single image using geometric reasoning on detected line segment produced from planar surfaces [24]. Recently, Zaheer *et al*. used orthogonal lines in a multi planar single view, to recover full 3D reconstruction of the scene [43]. These approaches are particularly interesting for indoor man-made environments or for outdoor scenes with man made structures around. The reason is that human architecture involves a profusion of parallel and orthogonal lines emanating from planar surfaces.

Apart from 3D model recovery from a single view, substantial work has also been done in single view metrology. Leibowitz and Zisserman proposed metric rectification of planes from single perspective view and used it to measure angles, lengths and ratios under given constraints of one known angle, two unknown but equal angles and a known length ratio [27]. Zaheer and Khan showed that projectively distorted views of man-made 3D structures can be metric rectified using orthogonal plane-pairs [42]. Liu *et al*. estimated depth from a single view given semantic segmentation under geometric constraints [28]. On this topic, Criminisi *et al*. explain single and multi view metric reconstructions in detail, especially determining camera pose, computing distance between planes parallel to a reference plane and also finding area and length ratios on any plane parallel to the reference plane [7, 6].

### 2.2. Data-driven depth prediction

Digressing from a geometric approach, data-driven depth prediction techniques have used both supervised and unsupervised learning for stereo as well as monocular images. Sinz *et al*. used stereo images in Gaussian process re-



Authorized licensed use limited to: San Jose State University. Downloaded on February 05,2025 at 16:28:43 UTC from IEEE Xplore. Restrictions apply.

gression to estimate pixel wise depth map avoiding complications of stereo calibration [35]. Saxena *et al*. used monocular images of a variation of unstructured outdoor scenes for training Markov Random Fields (MRFs) on both local and global features for the recovery of fairly accurate depth maps [34]. Ladicky *et al*. designed a stereo matching classifier for predicting likelihood of matching pixel in one image offset by disparity in second image [23].

Owing to little success, the data-driven approaches for depth estimation could not acquire a lot of attention until the advent of deep learning. Deep learning based depth estimation made a huge impact in terms of holistic and accurate depth prediction so much so that a significant number of contributions have surfaced over a short span of time [40, 46, 25, 41, 45]. Žbontar and LeCunn trained a small convolutional neural network for matching stereo pairs instead of handcrafted feature matching and showed results better in terms of both the accuracy of depth maps as well as efficiency [44]. Eigen *et al*. used two deep networks in cascade [8] to predict high-level global depth followed by refinement to low-level local depth in a supervised regression framework. This approach acquired a lot of traction and several improvements followed such as adding Conditional Random Fields (CRFs) by Li *et al*. [39]. Cao *et al*. transformed monocular depth estimation from regression to a supervised classification problem [5]. Fu *et al*. used an ordinal regression loss in supervised training to eliminate slow convergence during training and sub-optimal minimas [10]. Their method is the best performing one on Kitti benchmark [2] with iRMSE 12.98 (1/km), RMSE 2.6 (m).

Supervised learning requires ground truth depth data which in case of deep learning becomes challenging to acquire. Therefore, recently, some unsupervised approaches have appeared that have performed exceptionally well. Garg *et al*. proposed a convolutional encoder network in which they use one of the stereo-pair style images and predict a disparity map [11]. An inverse warp of the second image is then produced from the known inter-view displacement and the disparity map, which is then used to reconstruct the first image. The photometric error in the reconstruction defines the network loss. An improvement to this monocular depth learning from stereo pairs was introduced by Godard *et al*. by integrating left-right consistency check [13]. Kundu *et al*. employed unsupervised adversarial learning for depth prediction [22].

# 3. Our approach

We use single view geometry for distance estimation. As argued earlier, single view geometry depends on the perspective of the parallel or orthogonal lines and planes in the image. As lane lines are generally parallel, we exploit the parallelism for this matter. We start with unknown camera extrinsics because it is usually difficult to obtain this information as it may change from time to time. Instead, we propose an automated way of finding camera height using horizon through cross-ratios of parallel lines [15]. Determination of camera height completes camera initialization which only needs to be done once for each rigid camera assembly. Road plane is then reconstructed and the distance from other vehicles on the road can be calculated through simple math. A block diagram of our full pipeline is shown in Figure 2. For camera initialization, we assume a straight and flat road surface. Lane boundaries and vehicles can be detected using any off-the-shelf lane and vehicle detection algorithms. In the following section, we will now briefly describe lane and vehicle detectors that are used in evaluation for distance estimation in this paper.

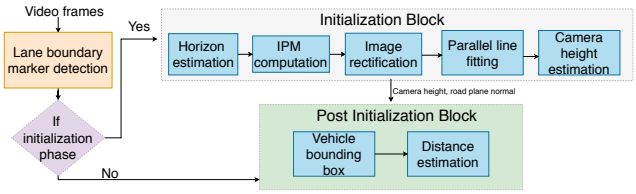



Figure 2. Block diagram of our algorithm.

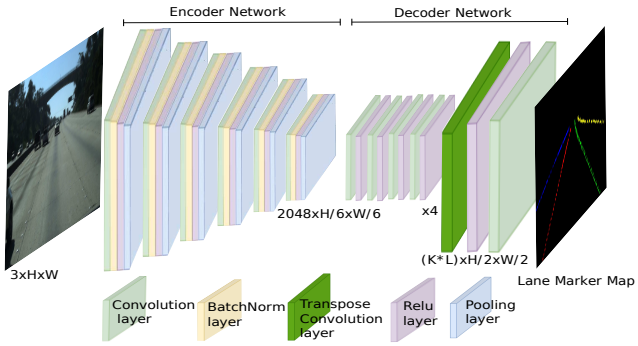
## 3.1. Lane and vehicle detection

To exploit road geometry, detection of lane boundaries is a crucial step for which we used the lane detection network by Khan *et al*. [20] which uses a fully convolutional deep network for detecting keypoints sampled along the lane boundaries. The network architecture diagram is shown in Figure 3(a). It consists of an encoder block (ResNet50 [16] in this case), responsible for feature extraction, and a decoder block that learns the relationship among extracted features and makes decision on top of it. The decoder block comprised of four convolutional layers, a single transpose-convolution layer (that upsamples feature map by 4x) followed by the output convolution layer. In the output layer, just like Mask-RCNN, one-hot mask for each keypoint is generated, where only the keypoint pixel is encoded (marked high or foreground).

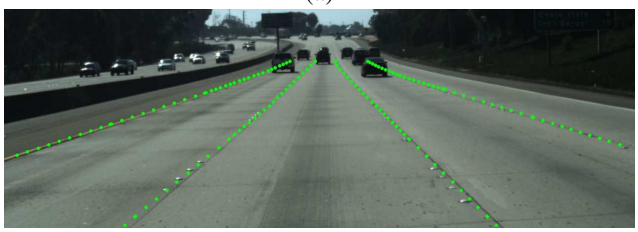
For training the lane detection network, CULane[31] dataset has been used. Every lane boundary ground truth was sampled into 40 keypoints, this number was chosen based on the best possible trade-off between efficiency and accuracy [20], with greater density of points closer to the road plane horizon (Figure 3(b)). This is done by splitting every lane into three vertical segments and sampling equally spaced keypoints (20, 10 and 10) from each segment starting from the one closer to the horizon. At inference time, line/curve is fitted on the predicted keypoints using RANSAC [9]. A best fit line must have a minimum number of inlier markers, which in case of 40 markers/lane boundary were kept at 10.



Authorized licensed use limited to: San Jose State University. Downloaded on February 05,2025 at 16:28:43 UTC from IEEE Xplore. Restrictions apply.




(a)



(b)

Figure 3. (a) The architecture of our lane detection network. The encoder is ResNet50 and the decoder consists of 4 convolution layers, 1 transpose-convolution layer followed by a fully convolutional output layer with one channel for each keypoint (b) The input image and keypoints sampled from the ground truth lane line are overlayed on the image. Number of keypoints increase towards the vanishing point.

For vehicle localization, any detection framework can be used. We fine-tuned COCO pre-trained Faster-RCNN model [33] with Resnet-50 backbone on Kitti vehicle dataset [12].

## 3.2. Camera initialization

Camera initialization involves finding camera height and road plane normal. This is our first contribution in this paper as along with camera intrinsics, we use road geometry and lane width which has not been used before in this regard. Integrating these known priors makes the process of initialization self-sufficient and fully automatic. It's a one-off calculation for every camera and stays constant until the camera is displaced or replaced altogether. It involves the following sub-modules:

### 3.2.1 Inverse Perspective Mapping

Inverse Perspective Mapping (IPM) is used to remove the perspective distortion in an image. Application of an IPM can synthetically rotate the camera into a top (bird's eye) view thus rendering the intersecting lines as parallel; the way they actually are in the real-world. This is required in order to find the camera height which is required for distance estimation. Therefore, we introduce a novel, accurate

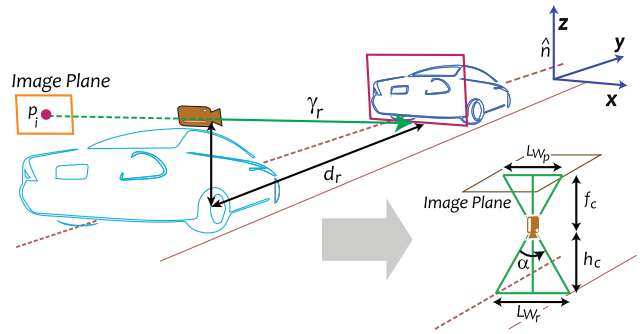



Figure 4. (Left) Point $p_i$ from the bottom of the detected vehicle is back projected to a ray $\gamma_r$ intersecting road plane at a point which gives the distance $d_r$ of the point from the camera on road plane with normal vector $\hat{n}$ (Right) Synthetically rotated camera view to estimate camera height $h_c$ given focal length $f_c$, lane width in rectified image $Lw_p$ and lane width in real world $Lw_r$

and robust way of estimating an IPM by exploiting road geometry with no prior knowledge of the camera pose.

Our approach is aimed for a forward looking dashcam installed inside the driver's cabin. Configuration was done in a right-handed frame of reference. The road plane normal $\hat{n}$ was along the Z-axis (upwards), Y-axis was in the driving direction and X-axis towards the right side of the driver as shown in Figure 4(Left).

The estimation of IPM used a planar road scene having no elevation and more than one lane (3 or more lane boundaries), such as a typical highway scenario (Figure 5). Critical part of initialization was the estimation of a vanishing line or horizon which required at least three parallel lines in the rectified image.

### 3.2.2 Horizon estimation

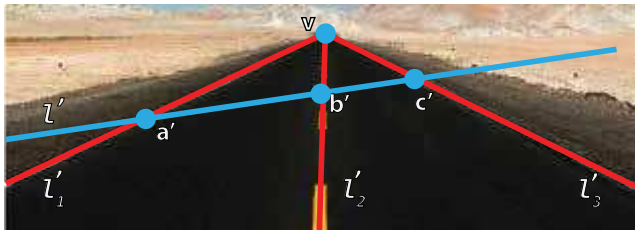
The horizon must be known to estimate the required camera rotation for an IPM. A horizon is a vanishing line of a plane which, in our case, is the road. In order to find vanishing line, we need two vanishing points. The first one can be simply calculated through the cross product of the vectors of any two lane lines expressed in homogeneous coordinates (see Figure 5 (a)). We call it the forward vanishing point (**V**). Finding a second vanishing point, however, is a bit involved.

According to Hartley and Zisserman Chapter 2 [14], if there are three co-linear points in an image $a'$, $b'$ and $c'$ (as shown in Figure 5 (a)) with known real world length ratios ($\frac{ab}{bc}$) of their corresponding real world points ($a$, $b$ and $c$) as shown in Figure 5 (b), we can find a point on the vanishing line of the plane on which these points lie. Lets call this point the lateral vanishing point or $V'$.

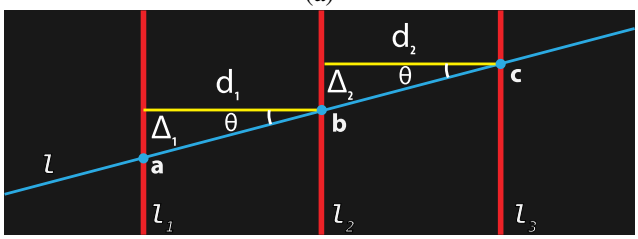
We take this concept and map it to our problem i.e. the road plane. For this, we will need three co-linear points existing on the road plane having known real world length



Authorized licensed use limited to: San Jose State University. Downloaded on February 05,2025 at 16:28:43 UTC from IEEE Xplore. Restrictions apply.




(a)




(b)

Figure 5. (a) Perspective view of road plane. Lane lines ($l_1'$, $l_2'$ and $l_3'$) are shown in red and an arbitrary line $l'$ (blue) cuts all lane lines at points $a'$, $b'$ and $c'$. Forward vanishing point (**V**) is obtained by the cross product of any two lane lines (b) Real world top-down view of the road plane. $d_1$ and $d_2$ are the distances between lane lines $l_1, l_2$ and $l_2, l_3$ respectively. Since all lane lines are parallel $\triangle_1$ and $\triangle_2$ are similar triangles which implies $\frac{ab}{bc} = \frac{d1}{d2}$

ratios. Since we already have detected lane lines in the image which lie on the road plane, we can find three co-linear points by drawing any line $l'$ which intersects all three lane lines at points $a'$, $b'$ and $c'$ respectively.

The points $a'$, $b'$ and $c'$ are in fact images of their real world counter parts $a$, $b$, and $c$ which lie on a line $l$ as shown in Figure 5 (b). The line $l'$ is the image of line $l$. In order to find the lateral vanishing point of the road plane, according to the description above, we need to know the length ratios between these real world points. The fact that these three points lie on three (parallel) lane lines ($l_1$, $l_2$ and $l_3$), it can be established through trigonometry that the ratio of the line segments $\frac{ab}{bc}$ is equal to $\frac{d_1}{d_2}$ where $d_1$ and $d_2$ are the distances between lines $l_1, l_2$ and $l_2, l_3$ respectively, as shown in figure (b). Here $d_1$ and $d_2$ are the lane widths.

Although the algorithm will still work if neighbouring lanes don't have same width (only requires ratio of the widths i.e. $\frac{d_1}{d_2}$ to be known), we assume the ratio to be 1 ($d_1$=$d_2$=lane width), since the lane lines are generally equidistant. Since the initialization is required only once for a camera, it can be done on a highway or a section of the highway where lanes are consistent/equidistant (using gps).

After getting two vanishing points (VPs) per frame, we collected all the sets of the vanishing points across all the frames of a video clip. The horizon was found as the best fitted line to the VPs as described in Algorithm 1.

**Algorithm 1:** Best fitting Horizon ($l_h$) estimation

**Input:** Detected markers by lane detector
**Output:** Estimated Horizon line

```
Find forward and lateral VP for each frame in a video:
initialize;
for i ≤ num_frames in video do
    Fit lines to keypoints (B = num_lines) ;
    if B ≥2 then
        for j <= ^B C_2 do
            forward_vpi_j = crossprod(line_1, line_2)
        end
    end
    if B ≥3 then
        for k <= ^B C_3 do
            Get lateral_vpi_k using cross ratios as described
              in Section 3.2.2
        end
    end
end
RANSAC for best fitting horizon across video:
initialize;
s = random_int between 1 to num_forward_vps;
t = random_int between 1 to num_lateral_vps;
while iterations ≤ t OR num_inliers ≤ max_inliers do
    Fit line l_h to forward_vp(s) and lateral_vp(t);
    if dist(l_h,forward_vp(s)) ≤ threshold_s &
      dist(l_h,lateral_vp(t)) ≤ threshold_t then
        record and count inlier points
    end
end
```

### 3.2.3 Image rectification

Now that the horizon has been determined, the IPM could be found. We computed road plane normal $\hat{n}$ using camera intrinsic matrix $K$ and horizon $l_h$ [14] by $\hat{n} = K^T\, l_h$. The IPM is then computed using $H_{IPM} = KRK^{-1}$, where $R = [l_h \times \hat{n}; (l_h \times \hat{n}) \times \hat{-n}; \hat{-n}]$ is the rotation matrix and $H_{IPM}$ is the rectification homography which will rotate the camera view to align its Z-axis with the road's normal as shown in Figure 4(Right). Frames could now be simply rectified into a bird's eye view by applying this IPM (Figure 4(Right)).

### 3.2.4 Parallel line fitting

Parallel lines can be represented as:

$$a_1x + b_1y + c_1 = 0,\ a_2x + b_2y + c_2 = 0$$

where the co-efficients $a_1 = a_2$ and $b_1 = b_2$ and only the intercept $c_1$ and $c_2$ are different. After a minimum number of inlier constraint is satisfied, parallel lines are fitted to the inlier markers using least squares by solving $Ax = 0$.

Here $A$ contains the inlier points (subscript) for each parallel line (superscript) in the rectified view. This system is solved using singular value decomposition for the least singular value.



Authorized licensed use limited to: San Jose State University. Downloaded on February 05,2025 at 16:28:43 UTC from IEEE Xplore. Restrictions apply.

Figure 6. Sample results on Kitti [12] (top) and nuScenes [3] (bottom). Displayed numbers are the distances in meters. Vehicles are numbered for clarity.

#### 3.2.5 Lane Width Initialization

In the rectified view of a given frame, a pair of consecutive parallel lines were selected and the distance between the lines was calculated per frame through:

$$L_{Wp} = \frac{\sum_f \frac{|c_2 - c_1|}{\sqrt{a^2+b^2}}}{f}. \tag{1}$$

where $f$ is the number of inlier frames and $L_{Wp}$ is the average initialized lane width. A frame is considered an inlier frame if it has at least one forward and one lateral vanishing point.

#### 3.2.6 Camera height estimation

Now that we have lane width in pixels $L_{Wp}$, we can use the camera focal length ($f_c$) to compute viewing angle $\alpha$ alpha using trigonometry

$$\alpha = 2 \times tan^{-1}(\frac{L_{Wp}/2}{f_c}) \tag{2}$$

Once camera viewing angle $\alpha$ is known, camera height $h_c$ is given by

$$h_c = \frac{\frac{L_{Wr}}{2}}{tan(\frac{\alpha}{2})} \tag{3}$$

Here $L_{Wr}$ is the lane width in real world and considering the fact that lane widths are usually standard across highways, $Lw_r$ is generally known. Determining camera height completes camera initialization. This is depicted in Figure 4 (Right).

### 3.3. Distance estimation

Once the camera is initialized, we know the camera height and road plane normal. We can then reconstruct the road plane:

$$\pi_r = \begin{bmatrix} \hat{n} \\ -h_c \end{bmatrix} \tag{4}$$

where $\hat{n}$ is the road plane normal and $\pi_r$ is $4 \times 1$ reconstructed road plane vector. A ray from the point $p_i$ in the image plane of the camera is given by $\gamma_r = K^{-1}p_i$. So the distance from the image plane to point where this ray intersects the reconstructed road plane in the real-world can be found by solving:

$$\pi_r^T \begin{bmatrix} d_r\gamma_r \\ 1 \end{bmatrix} = 0 \tag{5}$$

where $d_r$ is the distance where $\gamma_r$ intersects the plane:

$$d_r = \frac{h_c}{\hat{n} \cdot \gamma_r} \tag{6}$$

This is depicted in Figure 4 (Left). It is this simple arithmetic calculation that estimates the distance of the detected objects on the reconstructed road plane which makes it computationally inexpensive. Shift in the horizon which can be caused by changing slope of the road can impart difference to the observed distances. The best way to investigate the extent of error is to evaluate our algorithm on publicly collected datasets and compare them against the computationally expensive but state-of-the-art deep learning approaches.

## 4. Experimental setup

To evaluate our distance estimation algorithm, we used the well known datasets i.e. KITTI [12], nuScenes [3] and the recently released Lyft level 5 dataset [18]. All the datasets have color images along with Lidar data which is used as distance ground truth. nuScenes also provide front Radar data. In order to localize vehicles in the scenes, we used our vehicle detector as described in section 3.1. During the assessment, among the overlapping vehicle boxes only



Authorized licensed use limited to: San Jose State University. Downloaded on February 05,2025 at 16:28:43 UTC from IEEE Xplore. Restrictions apply.

| Algorithm | $\delta < 1.25$ | $\delta < 1.25^2$ | $\delta < 1.25^3$ | Abs.Rel | RMSE | RMSE log | Sq.Rel |
|---|---|---|---|---|---|---|---|
| (a) KITTI dataset | | | | | | | |
| DORN[10] | **0.92** | 0.96 | 0.98 | 0.11 | **2.44** | 0.18 | 0.44 |
| MonoDepth[13] | 0.82 | 0.94 | 0.98 | 0.13 | 6.34 | 0.21 | 1.36 |
| Ours | 0.53 | 0.89 | 0.98 | 0.29 | 6.24 | 0.30 | 2.02 |
| (b) nuScenes dataset | | | | | | | |
| DORN[10] | 0.55 | 0.85 | 0.96 | 0.25 | 7.43 | 0.33 | 1.83 |
| MonoDepth[13] | 0.56 | 0.81 | 0.93 | 0.23 | 8.38 | 0.36 | 2.16 |
| Radar | 0.73 | 0.80 | 0.85 | 0.48 | 11.66 | 0.71 | 13.25 |
| Ours | **0.78** | 0.94 | 0.97 | 0.15 | **6.10** | 0.24 | 1.29 |
| (c) Lyft Level 5 dataset | | | | | | | |
| DORN[10] | 0.11 | 0.30 | 0.67 | 0.73 | 11.43 | 0.57 | 7.91 |
| MonoDepth[13] | 0.03 | 0.12 | 0.45 | 0.50 | 12.23 | 0.77 | 5.50 |
| Ours | **0.69** | 0.87 | 0.93 | 0.17 | **7.31** | 0.32 | 1.53 |

Table 1. Distance estimation results on (a) KITTI (b) nuScenes (c) Lyft Level 5 datasets. Metrics in Red means higher is better. Metrics are explained in [8, 13]. Metrics in Blue means lower is better. Best scores are in bold. The scores of our algorithm are underlined.

(a)

(b)

Figure 7. Sample images from nuScenes [3]. Displayed numbers are the distances in meters. Sub-optimal horizon as compared to Figure 1 due to (a) downwards slope and (b) upwards slope. Recalculation required in this case.

the foremost box was selected. Hence only the un-occluded vehicles were used. Maximum distance cap of 70*m* was set since some lidars do not perform at larger distances. Besides, 70*m* braking distance provides enough room for a fair assessment.

Distance was in meters and we used the metrics defined in [8, 13], namely, Ratio threshold $\delta$, Absolute Relative difference (Abs Rel), Squared Relative difference (Sq Rel), RMSE (linear) and RMSE log for evaluation. Apart from our algorithm which is based on single view geometry, we also compared distance estimates of a supervised deep learning based method, DORN by Fu *et al*. [10] which has shown the best performance on Kitti depth estimation benchmark [2]. We used their model which is trained on the KITTI data for color images with lidar as ground truth. From unsupervised deep learning methods, we use MonoDepth by Godard *et al*. [13]. We used the model shared by the authors which is trained on Kitti's stereo data. MonoDepth predicts disparity which is converted into depth using the stereo configuration used in the data acquisition [Depth in meters = (FocalLength $\times$ Baseline in meters) / Disparity]. Lidar and radar data were projected on the image plane using camera matrix to produce depth maps. The vehicle depth was computed by averaging all the non zero depth values inside the vehicle box for lidar, MonoDepth and DORN. The detected box size was reduced by 50% in height and width with the same center in order to reduce the influence of the background. For radar however, we took the non zero minimum value inside the full sized box due to the sparsity of the data (works best for radar). For our algorithm, we took the middle point of the bottom of the full sized box for distance estimation because our algorithm requires the point to be on the road plane.

## 4.1. Results and Discussion

### 4.1.1 Tests on KITTI

KITTI dataset includes images from two color 1.4 MP PointGrey Flea2 cameras (left one is used in these experiments) and a Velodyne HDL-64E lidar data. Lidar data was mapped to the cameras. It does not have radar data. We used its six video clips which have the lidar as well as the corresponding video data in the KITTI validation split.

Table 1 (a) show the results and Figure 6 top-row shows some sample KITTI results. Based on the RMSE, we can see that supervised depth estimation by DORN performed much better than either MonoDepth or our algorithm. Both data driven approaches, DORN and MonoDepth were trained on the training and quiet possibly the validation sets of KITTI before the authors submitted the results to the KITTI test set benchmark [2]. Please note that the KITTI data was recorded by driving around the city of Karlsruhe, Germany, in the rural areas as well as on the highways [38]. And although our algorithm fell behind DORN in performance, still it had lower RMSE than the un-



Authorized licensed use limited to: San Jose State University. Downloaded on February 05,2025 at 16:28:43 UTC from IEEE Xplore. Restrictions apply.

supervised deep learning model (MonoDepth). Therefore, even after aiming for a highway driving scenario, it is still competing the state-of-the-art data driven approaches on a variety of road types. For a fair assessment of robustness and domain in-variance of the algorithms in consideration, we tested these models along with our algorithms on some other public datasets.

#### 4.1.2 Tests on nuScenes

nuScenes dataset contains recorded videos of mainly city scenes. It has 1000 video clips out of which 150 comprise the validation set that we used in this assessment. The dataset contains clips from multiple cameras installed on the vehicle. We used the front mounted camera CAM_FRONT for evaluation. Table 1 (b) show the results.

The results show that unlike KITTI, the deep learning networks both supervised and unsupervised, do not perform as well. Our algorithm however, performs better than all including radar in both $\delta$ (0.78) and RMSE (6.10). Figure 6 bottom row shows some results of nuScenes where horizon was estimated correctly and our algorithm performed well. In contrast, Figure 7 display example with sub-optimal horizon estimation and hence incorrect distance estimate which is a limitation of our system.

#### 4.1.3 Tests on Lyft Level 5 Dataset

Lyft dataset is the newest and the largest of all datasets. The setup is similar to nuScenes. We used the data from the forward looking Cam0 which includes 22680 sample images and collected on relatively planar roads as opposed to nuScenes making it the largest dataset in our evaluation setup. Table 1 (c) show the results. Since Monodepth and DORN are trained on KITTI, their performance is compromised on a new camera and its settings with RMSE of 12.23 and 11.43, respectively. Our algorithm, however, outperforms with an RMSE of 7.31.

### 4.2. Processing efficiency

In USA, the traffic rules require 3 to 5 secs headway from the vehicle in the front. Therefore, for an ADAS system, real-time operation is imperative as the response time is very short and compute power onboard is very limited. As argued earlier, this is one of the reasons for using lidars and radars. In this regard, we compared our algorithm with others for the processing time and the results are shown in Table 2. Deep networks require high compute power. As far as our method is concerned, there is no specific processing requirements. It is just a single division and a dot product which takes $2.31e^{-05}$ secs on Core i7 CPU. The initialization part can use any available Lane detection framework which will be in operation anyway within an ADAS or self-driving car. Our lane detection network discussed in section 3.1 runs at 0.125 $secs/frame$. But it's a one-off calculation for a limited number of frames and re-Initialization will not be required until the camera configuration is disturbed. This brings our algorithm at par with radar and lidar data processing without demanding any excessive compute power on onboard edge devices.

| Algorithm | DORN | MonoDepth | Ours |
|---|---|---|---|
| GPU | 2.10 | 0.124 | — |
| CPU | 130 | 0.795 | $23e^{-06}$ |

Table 2. Processing time per frame in secs averaged over 100 frames of nuScenes. GPU: Nvidia GTX-1080 with 8 GB on chip RAM, CPU:Intel Core i7 7820HK 2.9 GHz with 32 GB system RAM. Our algorithm's time is post initialization run-time for distance calculation. During initialization, our lane detector was also used at 0.124 secs/frame (one-time execution per camera).

## 5. Conclusion

In this paper we have presented a single view geometry based relative distance estimation algorithm for road vehicles. We have compared our algorithm with data driven deep learning (DL) based methods. Supervised deep learning performed exceptionally well on KITTI data. The compromise of the DL based algorithms on nuScenes and Lyft dataset was perhaps due to domain variance since the models were trained on KITTI data only. Therefore, although data driven approaches are the future, they do have their short comings. We instead proposed a geometry based alternative that is simple, intuitive, independent of data and performs almost equally on all the tested datasets keeping RMSE within a very narrow range i.e. 6.10 to 7.31. Our algorithm uses ADAS module of lane detection only for the initialization part which is computationally expensive since it's usually a DL based method. But initialization is a one-off calculation and after that, our distance estimation works real-time.

In this work, we also conclude that both geometric as well as deep learning frameworks are strong contestants to replace active sensors for ADAS systems as well as self-driving cars which will be a great help in reducing the cost of the intelligent transport systems of the future.

## Acknowledgements

This work was funded by KeepTruckin Inc. Authors greatly acknowledge the assistance of Mr. Numan Sheikh and Mr. Abdullah Hasan in the data collection and review process.

## References

[1] https://www.businessinsider.com/googles-waymo-reduces-lidar-cost-90-in-effort-to-scale-self-driving-cars-2017-1. *Business Insider, Google Waymo reduces cost of Lidars by 90%*, 2017.

[2] http://www.cvlibs.net/datasets/kitti/eval_depth.php? benchmark=depth_prediction. *KITTI Depth prediction benchmark*.



Authorized licensed use limited to: San Jose State University. Downloaded on February 05,2025 at 16:28:43 UTC from IEEE Xplore. Restrictions apply.

[3] https://www.nuscenes.org/. *NUScenes*, 2018.

[4] O. Barinova, V. Konushin, A. Yakubenko, K. Lee, H. Lim, and A. Konushin. Fast Automatic Single-View 3-d Reconstruction of Urban Scenes. In *European Conference on Computer Vision*, 2008.

[5] Y. Cao, Z. Wu, and C. Shen. Estimating Depth From Monocular Images as Classification Using Deep Fully Convolutional Residual Networks. *IEEE Transactions on Circuits and Systems for Video Technology*, 28(11):3174–3182, nov 2018.

[6] A. Criminisi. *Accurate Visual Metrology from Single and Multiple Uncalibrated Images*. Springer-Verlag, Berlin, Heidelberg, 2001.

[7] A. Criminisi, I. Reid, and A. Zisserman. Single View Metrology. *International Journal of Computer Vision*, 40(2):123–148, nov 2000.

[8] D. Eigen, C. Puhrsch, and R. Fergus. Depth Map Prediction from a Single Image using a Multi-Scale Deep Network. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, 2014.

[9] M. A. Fischler and R. C. Bolles. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Communications of the ACM*, 24(6):381–395, jun 1981.

[10] H. Fu, M. Gong, C. Wang, K. Batmanghelich, and D. Tao. Deep Ordinal Regression Network for Monocular Depth Estimation. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2018.

[11] R. Garg, G. VijayKumarB., and I. D. Reid. Unsupervised CNN for Single View Depth Estimation: Geometry to the Rescue. In *European Conference on Computer Vision*, 2016.

[12] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The KITTI dataset. *International Journal of Robotics Research*, 32:1231–1237, 2013.

[13] C. Godard, O. Mac Aodha, and G. J. Brostow. Unsupervised Monocular Depth Estimation with Left-Right Consistency. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[14] R. Hartley and A. Zisserman. More Single View Geometry. In *Multiple view geometry in computer vision*, chapter 8, pages 205,216–219. Cambridge Univ. Press, 2nd edition, 2003.

[15] R. Hartley and A. Zisserman. Projective Geometry and Transformation of 2D. In *Mutiple View Geometry in Computer Vision*, chapter 2, pages 51–52. Cambridge University Press, 2nd edition, 2003.

[16] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[17] D. Hoiem, A. A. Efros, and M. Hebert. Geometric context from a single image. In *10th IEEE International Conference on Computer Vision*, volume 1, pages 654–661, oct 2005.

[18] V. Kesten, R. and Usman, M. and Houston, J. and Pandya, T. and Nadhamuni, K. and Ferreira, A. and Yuan, M. and Low, B. and Jain, A. and Ondruska, P. and Omari, S. and Shah, S. and Kulkarni, A. and Kazakova, A. and Tao, C. and Platinsky, L. and Jiang, W. and. Lyft Level 5 Dataset. *https://level5.lyft.com/dataset/*, 2019.

[19] M. Khader and S. Cherian. An Introduction to Automotive LIDAR (Texas Instruments). Technical report, 2018.

[20] H. Khan, A. Rafaqat, A. Hassan, A. Ali, W. Kazmi, and A. Zaheer. Lane detection using lane boundary marker network under road geometry constraints. In *IEEE Winter Conference on Applications of Computer Vision, Snowmass, Co)*. IEEE, 2020.

[21] V. Kolmogorov and R. Zabih. Multi-camera Scene Reconstruction via Graph Cuts. In *Proceedings of the 7th European Conference on Computer Vision-Part III*, ECCV '02, pages 82–96, London, UK, 2002. Springer-Verlag.

[22] J. N. Kundu, P. K. Uppala, A. Pahuja, and R. V. Babu. AdaDepth: Unsupervised Content Congruent Adaptation for Depth Estimation. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2018.

[23] L. Ladicky, C. Häne, and M. Pollefeys. Learning the Matching Function. *CoRR*, abs/1502.0, 2015.

[24] D. C. Lee, M. Hebert, and T. Kanade. Geometric reasoning for single image structure recovery. *EEE Conference on Computer Vision and Pattern Recognition*, pages 2136–2143, 2009.

[25] Z. Li and N. Snavely. MegaDepth: Learning Single-View Depth Prediction from Internet Photos. In IEEE, editor, *IEEE Computer Vision and Pattern Recognition*, pages 2041–2050, 2018.

[26] D. Liebowitz, A. Criminisi, and A. Zisserman. Creating Architectural Models from Images. *Computer Graphics Forum*, 18(3):39–50, 1999.

[27] D. Liebowitz and A. Zisserman. Metric rectification for perspective images of planes. In *International Conference on Computer and Pattern Recognition*. IEEE Comput. Soc, 1998.

[28] B. Liu, S. Gould, and D. Koller. Single image depth estimation from predicted semantic labels. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1253–1260, jun 2010.

[29] J. Oberhammer, N. Somjit, U. Shah, and Z. Baghchehsaraei. RF MEMS for automotive radar. In D. Uttamchandani, editor, *Handbook of Mems for Wireless and Mobile Applications*, Woodhead Publishing Series in Electronic and Optical Materials, chapter 16, pages 518–549. Woodhead Publishing, 2013.

[30] J. Pan, M. Hebert, and T. Kanade. Inferring 3D layout of building facades from a single image. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2918–2926, 2015.

[31] X. Pan, J. Shi, P. Luo, X. Wang, and X. Tang. Spatial As Deep: Spatial CNN for Traffic Scene Understanding. In *AAAI Conference on Artificial Intelligence*, pages 7276–7283, 2018.

[32] K.-Y. Park and S.-Y. Hwang. Robust Range Estimation with a Monocular Camera for Vision-Based Forward Collision Warning System. *The Scientific World Journal*, (923632):9, 2014.

[33] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In C. Cortes, N. D. Lawrence, D. D. Lee,



Authorized licensed use limited to: San Jose State University. Downloaded on February 05,2025 at 16:28:43 UTC from IEEE Xplore. Restrictions apply.

M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 91–99. Curran Associates, Inc., 2015.

[34] A. Saxena, S. H. Chung, and A. Y. Ng. Learning Depth from Single Monocular Images. In *Proceedings of the 18th International Conference on Neural Information Processing Systems*, NIPS'05, pages 1161–1168, Cambridge, MA, USA, 2005. MIT Press.

[35] F. H. Sinz, J. Q. Candela, G. H. Bakir, C. E. Rasmussen, and M. O. Franz. Learning Depth from Stereo. In C. E. Rasmussen, H. H. Bülthoff, B. Schölkopf, and M. A. Giese, editors, *Pattern Recognition*, pages 245–252, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

[36] P. Sturm and S. Maybank. A Method for Interactive 3D Reconstruction of Piecewise Planar Objects from Single Images. In *British Machine Vision Conference*, pages 265–274, 1999.

[37] TuSimple. TuSimple Velocity Estimation Challenge. pages https://github.com/TuSimple/tusimple–benchmark/tre, 2017.

[38] J. Uhrig, N. Schneider, L. Schneider, U. Franke, T. Brox, and A. Geiger. Sparsity Invariant CNNs. In *International Conference on 3D Vision (3DV)*, 2017.

[39] B. L. . C. S. . Y. D. . A. van den Hengel ; Mingyi He. Depth and surface normal estimation from monocular images using regression on deep features and hierarchical CRFs. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1119–1127, jun 2015.

[40] C. Wang, J. M. Buenaposada, R. Zhu, and S. Lucey. Learning Depth from Monocular Videos using Direct Methods. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2018.

[41] Z. Yin and J. Shi. GeoNet: Unsupervised Learning of Dense Depth, Optical Flow and Camera Pose. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[42] A. Zaheer and S. Khan. 3D Metric Rectification using Angle Regularity. *IEEE Winter Conference on Applications of Computer Vision*, pages 31–36, 2014.

[43] A. Zaheer, M. Rashid, M. A. Riaz, and S. Khan. Single-View Reconstruction using orthogonal line-pairs. *Computer Vision and Image Understanding*, 172:107–123, 2018.

[44] J. Žbontar and Y. LeCun. Stereo Matching by Training a Convolutional Neural Network to Compare Image Patches. *Journal of Machine Learning Research*, 17(1):2287–2318, jan 2016.

[45] H. Zhan, R. Garg, C. S. Weerasekera, K. Li, H. Agarwal, and I. Reid. Unsupervised Learning of Monocular Depth Estimation and Visual Odometry with Deep Feature Reconstruction. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[46] Y. Zhang and T. Funkhouser. Deep Depth Completion of a Single RGB-D Image. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2018.



Authorized licensed use limited to: San Jose State University. Downloaded on February 05,2025 at 16:28:43 UTC from IEEE Xplore. Restrictions apply.