# Joint Monocular 3D Vehicle Detection and Tracking

Hou-Ning Hu[1]*, Qi-Zhi Cai[2]*, Dequan Wang[3], Ji Lin[4]*,
Min Sun[1], Philipp Krähenbühl[5], Trevor Darrell[3], Fisher Yu[3]
[1]National Tsing Hua University [2]Sinovation Ventures AI Institute
[3]UC Berkeley [4]MIT [5]UT Austin

## Abstract

*Vehicle 3D extents and trajectories are critical cues for predicting the future location of vehicles and planning future agent ego-motion based on those predictions. In this paper, we propose a novel online framework for 3D vehicle detection and tracking from monocular videos. The framework can not only associate detections of vehicles in motion over time, but also estimate their complete 3D bounding box information from a sequence of 2D images captured on a moving platform. Our method leverages 3D box depth-ordering matching for robust instance association and utilizes 3D trajectory prediction for re-identification of occluded vehicles. We also design a motion learning module based on an LSTM for more accurate long-term motion extrapolation. Our experiments on simulation, KITTI, and Argoverse datasets show that our 3D tracking pipeline offers robust data association and tracking. On Argoverse, our image-based method is significantly better for tracking 3D vehicles within 30 meters than the LiDAR-centric baseline methods.*

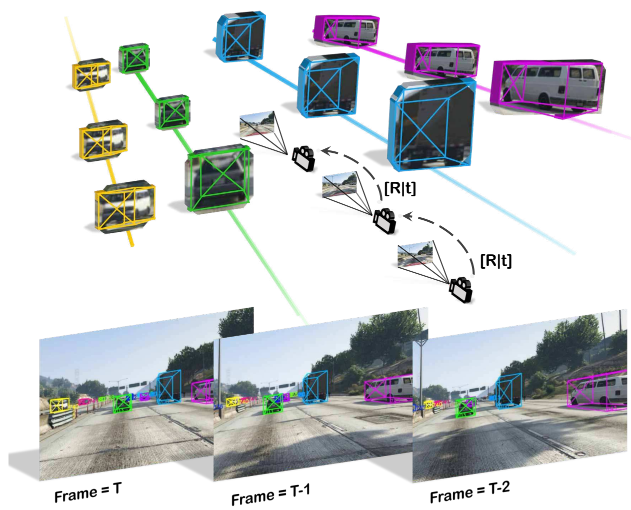


Figure 1: Joint online detection and tracking in 3D. Our dynamic 3D tracking pipeline predicts 3D bounding box association of observed vehicles in image sequences captured by a monocular camera with an ego-motion sensor.

## 1. Introduction

Autonomous driving motivates much of contemporary visual deep learning research. However, many commercially successful approaches to autonomous driving control rely on a wide array of views and sensors, reconstructing 3D point clouds of the surroundings before inferring object trajectories in 3D. In contrast, human observers have no difficulty in perceiving the 3D world in both space and time from simple sequences of 2D images rather than 3D point clouds, even though human stereo vision only reaches several meters. Recent progress in monocular object detection and scene segmentation offers the promise to make low-cost mobility widely available. In this paper, we explore architectures and datasets for developing similar capabilities using deep neural networks.

Monocular 3D detection and tracking are inherently ill-posed. In the absence of depth measurements or strong priors, a single view does not provide enough information to estimate 3D layout of a scene accurately. Without a good layout estimate, tracking becomes increasingly difficult, especially in the presence of large ego-motion (*e.g.*, a turning car). The two problems are inherently intertwined. Robust tracking helps 3D detection, as information along consecutive frames is integrated. Accurate 3D detection helps to track, as ego-motion can be factored out.

In this paper, we propose an online network architecture to jointly track and detect vehicles in 3D from a series of monocular images. Figure 1 provides an overview of our 3D tracking and detection task. After detecting 2D bounding boxes of objects, we utilize both world coordinates and re-projected camera coordinates to associate instances across frames. Notably, we leverage novel occlusion-aware associ-

*Work was done while Hou-Ning Hu, Qi-Zhi Cai and Ji Lin were at the Berkeley DeepDrive Center ation and depth-ordering matching algorithms to overcome the occlusion and reappearance problems in tracking. Finally, we capture the movement of instances in a world coordinate system and update their 3D poses using LSTM motion estimation along a trajectory, integrating single-frame observations associated with the instance over time.

Like any deep network, our model is data hungry. The more data we feed it, the better it performs. However, existing datasets are either limited to static scenes [41], lack the required ground truth trajectories [26], or are too small to train contemporary deep models [13]. To bridge this gap, we resort to realistic video games. We use a new pipeline to collect large-scale 3D trajectories, from a realistic synthetic driving environment, augmented with dynamic meta-data associated with each observed scene and object.

To the best of our knowledge, we are the first to tackle the estimation of complete 3D vehicle bounding box tracking information from a monocular camera. We jointly track the vehicles across frames based on deep features and estimate the full 3D information of the tracks including position, orientation, dimensions, and projected 3D box centers of each object. The depth ordering of the tracked vehicles constructs an important perceptual cue to reduce the mismatch rate. Our occlusion-aware data association provides a strong prior for occluded objects to alleviate the identity switch problem. Our experiments show that 3D information improves predicted association in new frames compared to traditional 2D tracking, and that estimating 3D positions with a sequence of frames is more accurate than single-frame estimation.

## 2. Related Works

**Object tracking** has been explored extensively in the last decade [44, 36, 39]. Early methods [4, 12, 21] track objects based on correlation filters. Recent ConvNet-based methods typically build on pre-trained object recognition networks. Some generic object trackers are trained entirely online, starting from the first frame of a given video [16, 1, 19]. A typical tracker will sample patches near the target object which are considered as foreground and some farther patches as background. These patches are then used to train a foreground-background classifier. However, these online training methods cannot fully utilize a large amount of video data. Held *et al*. [18] proposed a regression-based method for offline training of neural networks, tracking novel objects at test-time at 100 fps. Siamese networks also found in use, including tracking by object verification [40], tracking by correlation [3], tracking by detection [11]. Yu *et al*. [43] enhance tracking by modeling a track-let into different states and explicitly learns an Markov Decision Process (MDP) for state transition. Due to the absence of 3D information, it just uses 2D location to decide whether a track-let is occluded.

All those methods only take 2D visual features into consideration, where the search space is restricted near the original position of the object. This works well for a static observer, but fails in a dynamic 3D environment. Here, we further leverage 3D information to narrow down the search space, and stabilize the trajectory of target objects.

Sharma *et al*. [38] uses 3D cues for 2D vehicle tracking. Scheidegger *et al*. [37] also adds 3D kalman filter on the 3D positions to get more consistent 3D localization results. Because the goals are for 2D tracking, 3D box dimensions and orientation are not considered. Osep *et al*. [28] and Li *et al*. [22] studies 3D bounding box tracking with stereo cameras. Because the 3D depth can be perceived directly, the task is much easier, but in many cases such as ADAS, large-baseline stereo vision is not possible.

**Object detection** reaped many of the benefits from the success of convolutional representation. There are two mainstream deep detection frameworks: 1) two-step detectors: R-CNN [15], Fast R-CNN [14], and Faster R-CNN [31]. 2) one-step detectors: YOLO [29], SSD [24], and YOLO9000 [30].

We apply Faster R-CNN, one of the most popular object detectors, as our object detection input. The above algorithms all rely on scores of labeled images to train on. In 3D tracking, this is no different. The more training data we have, the better our 3D tracker performs. Unfortunately, getting a large amount of 3D tracking supervision is hard.

**Driving datasets** have attracted a lot of attention in recent years. KITTI [13], Cityscapes [8], Oxford RobotCar [25], BDD100K [47], NuScenes [5], and Argoverse [6] provide well annotated ground truth for visual odometry, stereo reconstruction, optical flow, scene flow, object detection and tracking. However, their provided 3D annotation is very limited compared to virtual datasets. Accurate 3D annotations are challenging to obtain from humans and expensive to measure with 3D sensors like LiDAR. Therefore these real-world datasets are typically small in scale or poorly annotated.

To overcome this difficulty, there has been significant work on virtual driving datasets: virtual KITTI [12], SYNTHIA [34], GTA5 [33], VIPER [32], CARLA [9], and Free Supervision from Video Games (FSV) [20]. The closest dataset to ours is VIPER [32], which provides a suite of videos and annotations for various computer vision problems while we focus on object tracking. We extend FSV [20] to include object tracking in both 2D and 3D, as well as fine-grained object attributes, control signals from driver actions.

In the next section, we describe how to generate 3D object trajectories from 2D dash-cam videos. Considering the practical requirement of autonomous driving, we primarily focus on online tracking systems, where only the past and current frames are accessible to a tracker.

## 3. Joint 3D Detection and Tracking

Our goal is to track objects and infer their precise 3D location, orientation, and dimension from a single monocular video stream and a GPS sensor. Figure 2 shows an overview

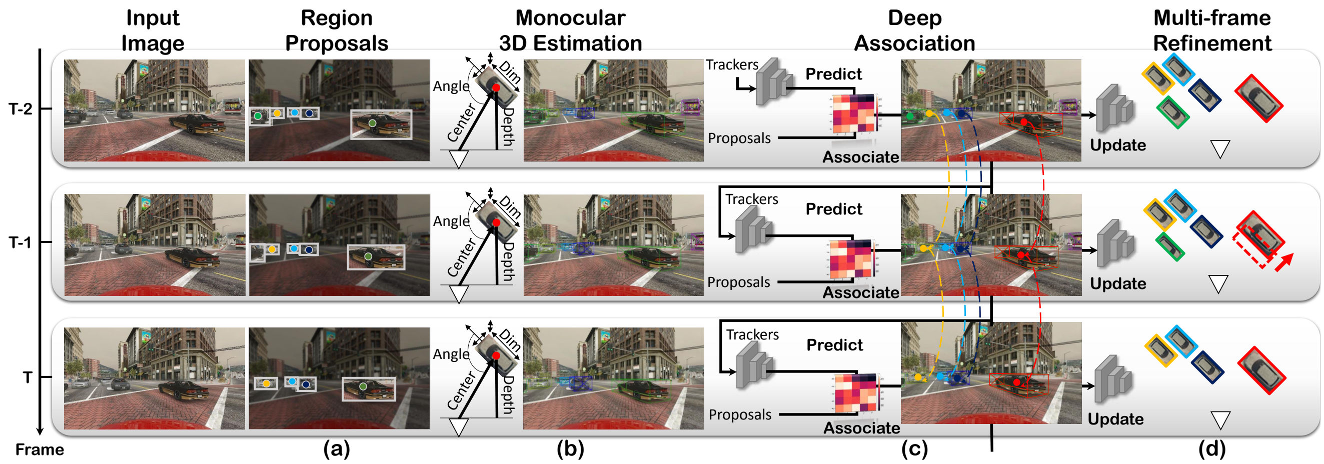


Figure 2: Overview of our monocular 3D tracking framework. Our online approach processes monocular frames to estimate and track region of interests (RoIs) in 3D (a). For each ROI, we learn 3D layout (*i.e*., depth, orientation, dimension, a projection of 3D center) estimation (b). With 3D layout, our LSTM tracker produces robust linking across frames leveraging occlusion-aware association and depth-ordering matching (c). With the help of 3D tracking, the model further refines the ability of 3D estimation by fusing object motion features of the previous frames (d).

of our system. Images are first passed through a detector network trained to generate object proposals and centers. These proposals are then fed into a layer-aggregating network which infers 3D information. Using 3D re-projection to generate similarity metric between all trajectories and detected proposals, we leverage estimated 3D information of current trajectories to track them through time. Our method also solves the occlusion problem in tracking with the help of occlusion-aware data association and depth-ordering matching. Finally, we re-estimate the 3D location of objects using the LSTM through the newly matched trajectory.

### 3.1. Problem Formulation

We phrase the 3D tracking problem as a supervised learning problem. We aim to find $N$ trajectories $\{\tau^1, \dots, \tau^N\}$, one for each object in a video. Each trajectory $\tau^i$ links a sequence of detected object states $\{s_a^{(i)}, s_{a+1}^{(i)}, \dots, s_b^{(i)}\}$ starting at the first visible frame $a$ and ending at the last visible frame $b$. The state of an object at frame $a$ is given by $s_a = (P, O, D, F, \Delta P)$, where $P$ defines the 3D world location $(x, y, z)$ of the object, and $\Delta P$ stands for its velocity $(\dot{x}, \dot{y}, \dot{z})$. $O, D, F$ denotes for object orientation $\theta$, dimension $(l, w, h)$ and appearance feature $f_{app}$, respectively. In addition, we reconstruct a 3D bounding box $X$ for each object, with estimated $P, O, D$ and the projection $c = (x_c, y_c)$ of 3D box's center in the image. The bounding boxes enable the use of our depth-ordering matching and occlusion-aware association. Each bounding box $X$ also forms a projected 2D box $M(X) = \{x_{\texttt{min}}, y_{\texttt{min}}, x_{\texttt{max}}, y_{\texttt{max}}\}$ projected onto a 2D image plane using camera parameters $M = K[R|t]$.

The intrinsic parameter $K$ can be obtained from camera calibration. The extrinsic parameter $[R|t]$ can be calculated from the commonly equipped GPS or IMU sensor. The whole system is powered by a convolutional network pipeline trained on a considerable amount of ground truth supervision. Next, we discuss each component in more detail.

### 3.2. Candidate Box Detection

In the paper, we employ Faster R-CNN [31] trained on our dataset to provide object proposals in the form of bounding boxes. Each object proposal (Figure 2(a)) corresponds to a 2D bounding box $\mathbf{d} = \{x_{\texttt{min}}, y_{\texttt{min}}, x_{\texttt{max}}, y_{\texttt{max}}\}$ as well as an estimated projection of the 3D box's center $c$. The detection results are used to locate the candidate vehicles and extract their appearance features. However, the centers of objects' 3D bounding boxes usually do not project directly to the center of their 2D bounding boxes. As a result, we have to provide an estimation of the 3D box center for better accuracy. More details about the estimation of the 3D center can be found in the supplementary material[1].

**Projection of 3D box center.** To estimate the 3D layout from single images more accurately, we extend the regression process to predict a projected 2D point of the 3D bounding box's center from an ROIpooled feature $F$ using L1 loss. Estimating a projection of 3D center is crucial since a small gap in the image coordinate will cause a gigantic shift in 3D. It is worth noting that our pipeline can be used with any off-the-shelf detector and our 3D box estimation module is extendable to estimate projected 2D points even if the detector is replaced. With the extended ROI head, the model

[1] Supplementary material of Joint Monocular 3D Vehicle Detection and Tracking can be found at https://eborboihuc.github.io/Mono-3DT/

regresses both a bounding box $\mathbf{d}$ and the projection of 3D box's center $c$ from an anchor point. ROIalign [17] is used instead of ROIpool to get the regional representation given the detected regions of interest (ROIs). This reduces the misalignment of two-step quantization.

### 3.3. 3D Box Estimation

We estimate complete 3D box information (Figure 2(b)) from an ROI in the image via a feature representation of the pixels in the 2D bounding box. The ROI feature vector $F$ is extracted from a 34-layer DLA-up [46] using ROIalign. Each of the 3D information is estimated by passing the ROI features through a 3-layer 3x3 convolution sub-network, which extends the stacked Linear layers design of Mousavian *et al*. [27]. We focus on 3D location estimation consisting of object center, orientation, dimension and depth, whereas [27] focus on object orientation and dimension from 2D boxes. Besides, our approach integrates with 2D detection and has the potential to jointly training, while [27] crops the input image with pre-computed boxes. This network is trained using ground truth depth, 3D bounding box center projection, dimension, and orientation values. A convolutional network is used to preserve spatial information. In the case that the detector is replaced with another architecture, the center $c$ can be obtained from this sub-network.

**3D World Location.** Contrasting with previous approaches, we also infer 3D location $P$ from monocular images. The network regresses an inverse depth value $1/d$, but is trained to minimize the L1 loss of the depth value $d$ and the projected 3D location $P$. A projected 3D location $P$ is calculated using an estimated 2D projection of the 3D object center $c$ as well as the depth $d$ and camera transformation $M$.

**Vehicle Orientation.** Given the coordinate distance $\hat{x} = x_c - \frac{w}{2}$ to the horizontal center of an image and the focal length $f$, we can restore the global rotation $\theta$ in the camera coordinate from $\theta_l$ with simple geometry, $\theta = (\theta_l + \arctan \frac{\hat{x}}{f}) \mod 2\pi$. Following [27] for $\theta_l$ estimation, we first classify the angle into two bins and then regress the residual relative to the bin center using Smooth L1 loss.

**Vehicle Dimension.** In driving scenarios, the high variance of the distribution of the dimensions of different categories of vehicles (e.g., car, bus) results in difficulty classifying various vehicles using unimodal object proposals. Therefore, we regress a dimension $D$ to the ground truth dimension over the object feature representation using L1 loss.

The estimation of an object's 3D properties provides us with an observation for its location $P$ with orientation $\theta$, dimension $D$ and 2D projection of its 3D center $c$. For any new tracklet, the network is trained to predict monocular object state $s$ of the object by leveraging ROI features. For any previously tracked object, the following association network is able to learn a mixture of a multi-view monocular 3D estimates by merging the object state from last visible frames and the current frame. First, we need to generate such a 3D trajectory for each tracked object in world coordinates.

### 3.4. Data Association and Tracking

Given a set of tracks $\{\tau^J, \dots, \tau^K\}$ at frame $a$ where $1 \leq J \leq K \leq M$ from $M$ trajectories, our goal is to associate each track with a candidate detection, spawn new tracks, or end a track (Figure 2(c)) in an online fashion.

We solve the data association problem by using a weighted bipartite matching algorithm. Affinities between tracks and new detections are calculated from two criteria: overlap between projections of current trajectories forward in time and bounding boxes candidates; and the similarity of the deep representation of the appearances of new and existing object detections. Each trajectory is projected forward in time using the estimated velocity of an object and camera ego-motion. Here, we assume that ego-motion is given by a sensor, like GPS, an accelerometer, gyro and/or IMU.

We define an affinity matrix $\mathbf{A}(\tau_a, \mathbf{s_a})$ between the information of an existing track $\tau_a$ and a new candidate $\mathbf{s_a}$ as a joint probability of appearance and location correlation.

$$\mathbf{A}_{\texttt{deep}}(\tau_a, \mathbf{s}_a) = \exp(-||F_{\tau_a}, F_{\mathbf{s}_a}||_1) \tag{1}$$

$$\mathbf{A}_{\texttt{2D}}(\tau_a, \mathbf{s}_a) = \frac{\mathbf{d}_{\tau_a} \cap \mathbf{d}_{\mathbf{s}_a}}{\mathbf{d}_{\tau_a} \cup \mathbf{d}_a} \tag{2}$$

$$\mathbf{A}_{\texttt{3D}}(\tau_a, \mathbf{s}_a) = \frac{\mathbf{M}(X_{\tau_a}) \cap \mathbf{M}(X_{\mathbf{s}_a})}{\mathbf{M}(X_{\tau_a}) \cup \mathbf{M}(X_{\mathbf{s}_a})}, \tag{3}$$

where $F_{\tau_a}$, $F_{\mathbf{s}_a}$ are the concatenation of appearance feature $f_{app}$, dimension $D$, center $c$, orientation $\theta$ and depth $d$. $X_{\tau_a}$ and $X_{\mathbf{s}_a}$ are the tracked and predicted 3D bounding boxes, $\mathbf{M}$ is the projection matrix casting the bounding box to image coordinates, $\mathbf{A}_{\texttt{2D}}$ and $\mathbf{A}_{\texttt{3D}}$ is the Intersection of Union (IoU).

$$\begin{aligned}\mathbf{A}(\tau_a, \mathbf{s}_a) = w_{\texttt{deep}} \mathbf{A}_{\texttt{app}}(\tau_a, \mathbf{s}_a) + w_{\texttt{2D}} \mathbf{A}_{\texttt{2D}}(\tau_a, \mathbf{s}_a) \\ + w_{\texttt{3D}} \mathbf{A}_{\texttt{3D}}(\tau_a, \mathbf{s}_a)\end{aligned} \tag{4}$$

$w_{\texttt{deep}}, w_{\texttt{2D}}, w_{\texttt{3D}}$ are the weights of appearance, 2D overlap, and 3D overlap. We utilize a mixture of those factors as the affinity across frames, similar to the design of POI [45].

Comparing to 2D tracking, 3D-oriented tracking is more robust to ego-motion, visual occlusion, overlapping, and re-appearances. When a target is temporally occluded, the corresponding 3D motion estimator can roll-out for a period of time and relocate 2D location at each new point in time via the camera coordinate transformation.

**Depth-Ordering Matching.** We introduce instance depth ordering for assigning a detection to neighbor tracklets, which models the strong prior of relative depth ordering found in human perception. For each detection of interest (DOI), we consider potential associated tracklets in order of their depths. From the view of each DOI, we obtain the IOU of two non-occluded overlapping map from both ascending

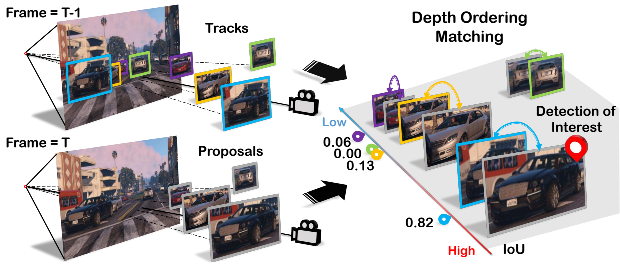


Figure 3: Illustration of depth-ordering matching. Given the tracklets and detections, we sort them into a list by depth order. For each detection of interest (DOI), we calculate the IOU between DOI and non-occluded regions of each tracklet. The depth order naturally provides higher probabilities to tracklets near the DOI.

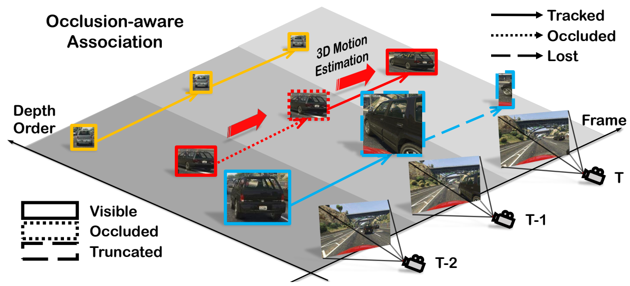


Figure 4: Illustration of Occlusion-aware association. A tracked tracklet (yellow) is visible all the time, while a tracklet (red) is occluded by another (blue) at frame $T-1$. During occlusion, the tracklet does not update state but keep inference motion until reappearance. For a truncated or disappear tracklet (blue at frame $T$), we left it as lost.

and descending ordering. To cancel out the ordering ambiguity of a distant tracklet, we filter out those tracklets with a larger distance to a DOI than a possible matching length. So Equation 3 becomes

$$\mathbf{A}_{\mathtt{3D}}(\tau_a, \mathbf{s}_a) = \mathbb{1} \times \frac{\phi(\mathbf{M}(X_{\tau_a})) \cap \mathbf{M}(X_{\mathbf{s}_a})}{\phi(\mathbf{M}(X_{\tau_a})) \cup \mathbf{M}(X_{\mathbf{s}_a})}, \quad (5)$$

where $\mathbb{1}$ denotes if the tracklets is kept after depth filtering, and the overlapping function

$$\phi(\cdot) = \arg\min_{x}\{x|\mathtt{ord}(x) < \mathtt{ord}(x_0) \forall x_0 \in \mathbf{M}(X_{\tau_a}))\}$$

captures pixels of non-occluded tracklets region with the nearest depth order. It naturally provides higher probabilities of linking neighbor tracklets than those layers away. In this way, we obtain the data association problem of moving objects with the help of 3D trajectories in world coordinates. Figure 3 depicts the pipeline of depth ordering. Finally, we solve data association using the Kuhn-Munkres algorithm.

**Occlusion-aware Data Association.** Similar to previous state-of-the-art methods [42, 43, 35], we model the lifespan of a tracker into four major subspaces in MDP state space: $\{\mathtt{birth}, \mathtt{tracked}, \mathtt{lost}, \mathtt{death}\}$. For each new set of detections, the tracker is updated using pairs with the highest affinities score (Equation 4). Each unmatched detection spawns a new tracklet; however, an unmatched tracklet is not immediately terminated, as tracklets can naturally disappear in occluded region and reappear later. We continue to predict the 3D location of unmatched tracklets until they disappear from our tracking range (*e.g.* $-10m$ to $100m$) or die out after 20 time-steps. We address the dynamic object inter-occlusion problem by separating a new state called "`occluded`" from a `lost` state. An object is considered `occluded` when covered by another object in the front with over $70\%$ overlap. An `occluded` tracklet will not update its lifespan or its feature representation until it is clear from occlusion, but we still predict its 3D location using the estimated motion. Figure 4 illustrates how the occlusion-aware association works. In the next subsection, we show how to estimate that distance leveraging the associated tracklet and bounding box using a deep network.

### 3.5. Motion Model

**Deep Motion Estimation and Update.** To exploit the temporal consistency of certain vehicles, we associate the information across frames by using two LSTMs. We embed a 3D location $P$ to a 64-dim location feature and use 128-dim hidden state LSTMs to keep track of a 3D location from the 64-dim output feature.

Prediction LSTM (P-LSTM) models dynamic object location in 3D coordinates by predicting object velocity from previously updated velocities $\dot{P}_{T-n:T-1}$ and the current possible location $\tilde{P}_T$. We use previous $n = 5$ frames of vehicle velocity to model object motion and acceleration from the trajectory. Given the current expected location of the object from 3D estimation module, the Updating LSTM (U-LSTM) considers both current $\hat{P}_T$ and previously predicted location $\tilde{P}_{T-1}$ to update the location and velocity (Figure 2(c)).

Modeling motion in 3D world coordinates naturally cancels out adverse effects of ego-motion, allowing our model to handle missed and occluded objects. The LSTMs continue to update the object state

$$\mathbf{s}_a^{(i)} = \mathbf{s}_{a-1}^{(i)} + \alpha(\mathbf{s}_a^* - \mathbf{s}_{a-1}^{(i)})$$

using the observation of matched detection state $\mathbf{s}_a^*$ with an updating ratio $\alpha = 1 - \mathbf{A}_{deep}(\tau_a^i, \mathbf{s}_a^*)$, while assuming a linear velocity model if there is no matched bounding box. Therefore, we model 3D motion (Figure 2(d)) in world coordinates allowing occluded tracklet to move along motion plausible paths while managing the birth and death of moving objects.

Concretely, our pipeline consists of a single-frame monocular 3D object detection model for object-level pose inference and recurrent neural networks for inter-frame object association and matching. We extend the region processing to include 3D estimation by employing multi-head modules for each object instance. We introduced occlusion-aware association to solve inter-object occlusion problem. For tracklet matching, depth ordering lowers mismatch rate by filtering out distant candidates from a target. The LSTM motion estimator updates the velocity and states of each object independent of camera movement or interactions with other objects. The final pipeline produces accurate and dense object trajectories in 3D world coordinate system.

# 4. 3D Vehicle Tracking Simulation Dataset

It is laborious and expensive to annotate a large-scale 3D bounding box image dataset even in the presence of LiDAR data, although it is much easier to label 2D bounding boxes on tens of thousands of videos [47]. Therefore, no such dataset collected from real sensors is available to the research community. To resolve the data problem, we turn to driving simulation to obtain accurate 3D bounding box annotations at no cost of human efforts. Our data collection and annotation pipeline extend the previous works like VIPER [32] and FSV [20], especially in terms of linking identities across frames. Details on the thorough comparison to prior data collection efforts, and dataset statistics can be found in the supplementary material.

Our simulation is based on *Grand Theft Auto V*, a modern game that simulates a functioning city and its surroundings in a photo-realistic three-dimensional world. To associate object instances across frames, we utilize in-game API to capture global instance id and corresponding 3D annotations directly. In contrast, VIPER leverages a weighted matching algorithm based on a heuristic distance function, which can lead to inconsistencies. It should be noted that our pipeline is real-time, providing the potential of large-scale data collection, while VIPER requires expensive off-line processing.

# 5. Experiments

We evaluate our 3D detection and tracking pipeline on Argoverse Tracking benchmark [6], KITTI MOT benchmark [13] and our large-scale dataset, featuring real-world driving scenes and a wide variety of road conditions in a diverse virtual environment, respectively.

## 5.1. Training and Evaluation

**Dataset.** Our GTA raw data is recorded at 12 FPS, which is helpful for temporal aggregation. With the goal of autonomous driving in mind, we focus on vehicles closer than $150m$, and also filtered out the bounding boxes whose areas are smaller than $256$ pixels. The dataset is then split into train, validation and test set with ratio $10 : 1 : 4$. The KITTI Tracking benchmark provides real-world driving scenario. Our 3D tracking pipeline train on the whole training set and evaluate the performance on the public testing benchmark. The Argoverse Tracking benchmark offers novel 360 degree driving dataset. We train on the training set and evaluate the performance on the validation benchmark since the evaluation server is not available upon the time of submission.

**Training Procedure.** We train our 3D estimation network (Section 3.3) on each training set, separately. 3D estimation network produces feature maps as the input of ROIalign [17]. The LSTM motion module (Section 3.5) is trained on the same set with a sequence of $10$ images per batch. For GTA, all the parameters are searched using validation set with detection bounding boxes from Faster R-CNN. The training is conducted for $100$ epochs using Adam optimizer with an initial learning rate $10^{-3}$, momentum $0.9$, and weight decay $10^{-4}$. Each GPU has $5$ images and each image with no more than $300$ candidate objects before NMS. More training details can be found in supplementary material.

**3D Estimation.** We adapt depth evaluation metrics [10] from image-level to object-level, leveraging both error and accuracy metrics. Error metrics include absolute relative difference (Abs Rel), squared relative difference (Sq Rel), root mean square error (RMSE) and RMSE log. Accuracy metrics are percents of $y_i$ that $\texttt{max}(\frac{y_i}{y_i^*}, \frac{y_i^*}{y_i}) < \delta$ where $\delta = 1.25, 1.25^2, 1.25^3$. Following the setting of KITTI [13], we use orientation score (OS) for orientation evaluation.

We propose two metrics for evaluating estimated object dimension and 3D projected center position. A Dimension Score ($DS$) measures how close an object volume estimation to a ground truth. $DS$ is defined as $DS = \min(\frac{V_{\texttt{pred}}}{V_{\texttt{gt}}}, \frac{V_{\texttt{gt}}}{V_{\texttt{pred}}})$ with an upper bound $1$, where $V$ is the volume of a 3D box by multiplying its dimension $l * w * h$. A Center Score ($CS$) measures distance of a projected 3D center and a ground truth. $CS$ is calculated by $CS = (1 + \cos(a_{\texttt{gt}} - a_{\texttt{pd}}))/2$, with a upper bound $1$, where $a$ depicts an angular distance $((x_{\texttt{gt}} - x_{\texttt{pd}})/w_{\texttt{pd}}, (y_{\texttt{gt}} - y_{\texttt{pd}})/h_{\texttt{pd}})$, weighted by corresponding box width and height in the image coordinates.

**Object Tracking.** We follow the metrics of CLEAR [2], including multiple object tracking accuracy (MOTA), multiple object tracking precision (MOTP), miss-match (MM), false positive (FP), and false negative (FN), etc.

**Overall Evaluation.** We evaluated the 3D IoU mAP of 3D layout estimation with refined depth estimation of different tracking methods. The metric reflects the conjunction of all 3D components, dimension, rotation, and depth.

## 5.2. Results

**3D for tracking.** The ablation study of tracking performance could be found in Table 1. Adding deep feature distinguishes two near-overlapping objects, our false negative (FN) rate drops with an observable margin. With depth-order

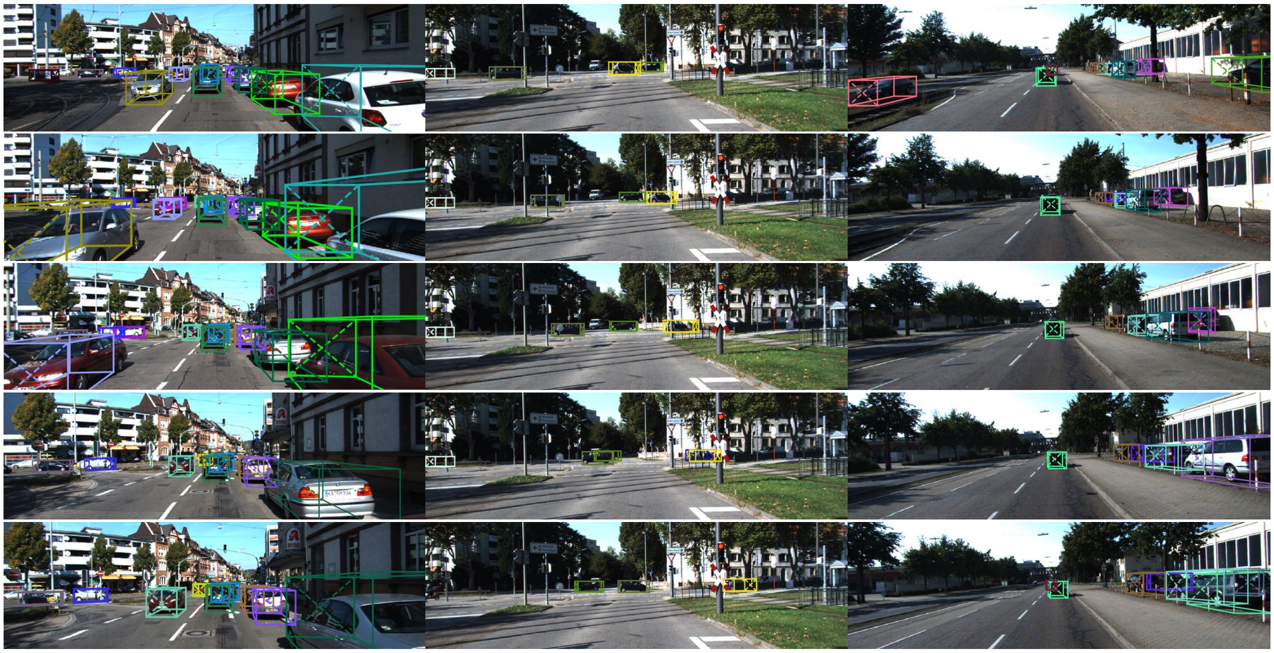

Figure 5: Qualitative results of 3D Estimation on KITTI testing set. We show predicted 3D layout colored with tracking IDs.

| Motion | Deep | Order | MOTA↑ | FN↓ | MM↓ | Ratio (%)↑ |
|---|---|---|---|---|---|---|
| KF3D | - | - | 69.616 | 21.298 | 17480 | 0 |
| KF3D | - | ✓ | 70.061 | 21.319 | **16042** | 8.222 |
| KF3D | ✓ | - | 69.843 | 18.547 | 27639 | 0 |
| KF3D | ✓ | ✓ | 70.126 | 18.177 | 25860 | 6.434 |
| LSTM | ✓ | ✓ | **70.439** | **18.094** | 23863 | 13.662 |

Table 1: Ablation study of tracking performance with different methods in GTA dataset. Motion column shows which motion model being used. KF stands for Kalman Filter. Ratio is the relative improvement in reducing the number of mismatch pairs. Using deep feature in correlation can reduce the false negative (FN) rate. Using depth-order matching and occlusion-aware association filter out relatively $6-8\%$ ($\frac{(17480-16042)}{16042} \times 100\%$) mismatching trajectories. LSTM modeling dynamic motion benefits 3D IoU AP in Table 2.

matching and occlusion-aware association, our model filters out $6-8\%$ possible mismatching trajectories. For a full ablation study, please refer to the supplementary material.

**Motion Modeling.** We propose to use LSTM to model the vehicle motion. To analyze its effectiveness, we compare our LSTM model with traditional 3D Kalman filter (KF3D) and single frame 3D estimation using 3D IoU mAP. Table 2 shows that KF3D provides a small improvement via trajectory smoothing within prediction and observation. On the other hand, our LSTM module gives a learned estimation based on past $n$ velocity predictions and current frame observation, which may compensate for the observation error. Our LSTM module achieves the highest accuracy among the

| Method | Easy $AP^{70}_{3d}$ | Easy $AP^{50}_{3d}$ | Easy $AP^{25}_{3d}$ | Medium $AP^{70}_{3d}$ | Medium $AP^{50}_{3d}$ | Medium $AP^{25}_{3d}$ | Hard $AP^{70}_{3d}$ | Hard $AP^{50}_{3d}$ | Hard $AP^{25}_{3d}$ |
|---|---|---|---|---|---|---|---|---|---|
| None | 6.13 | 35.12 | 69.52 | 4.93 | 24.25 | 53.26 | 4.05 | 17.26 | 41.33 |
| KF3D | 6.14 | 35.15 | 69.56 | 4.94 | 24.27 | 53.29 | 4.06 | 17.27 | 41.42 |
| LSTM | **7.89** | **36.37** | **73.39** | **5.25** | **26.18** | **53.61** | **4.46** | **17.62** | **41.96** |

Table 2: Comparison of tracking performance on 3D IoU AP 25, 50, 70 in GTA dataset. Noted that KF3D slightly improves the AP performance compare to single-frame estimation (None), while LSTM grants a clear margin. The difference comes from how we model object motion in the 3D coordinate. Instead of using Kalman filter smoothing between prediction and observation, we directly model vehicle movement using LSTMs.

| | IDS ↓ | FRAG ↓ | FP ↓ | FN ↓ | MOTA ↑ | MOTP ↑ |
|---|---|---|---|---|---|---|
| 2D Center | 315 | 497 | 401 | 1435 | 91.06 | 90.36 |
| 3D Center | **190** | **374** | **401** | **1435** | **91.58** | **90.36** |

Table 3: Importance of using projection of 3D bounding box center estimation on KITTI training set. We evaluate our proposed model using different center inputs $c$ to reveal the importance of estimating projection of a 3D center. The reduction of ID Switch (IDS), track fragmentation (FRAG), and the increase of MOTA suggest that the projection of a 3D center benefits our tracking pipeline over the 2D center.

other methods with all the IoU thresholds.

**3D Center Projection Estimation.** We estimate the 3D location of a bounding box through predicting the projection of its center and depth, while Mousavian *et al.* [27] uses the center of detected 2D boxes directly. Table 3 shows the

| Dataset | Amount | Depth Error Metric | | | | Depth Accuracy Metric | | | Orientation | Dimension | Center |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Abs Rel ↓ | Sq Rel ↓ | RMSE ↓ | RMSElog ↓ | $\delta < 1.25$ ↑ | $\delta < 1.25^2$ ↑ | $\delta < 1.25^3$ ↑ | OS ↑ | DS ↑ | CS ↑ |
| GTA | 1% | 0.258 | 4.861 | 10.168 | 0.232 | 0.735 | 0.893 | 0.934 | 0.811 | 0.807 | 0.982 |
| | 10% | 0.153 | 3.404 | 6.283 | 0.142 | 0.880 | 0.941 | 0.955 | 0.907 | 0.869 | 0.982 |
| | 30% | 0.134 | 3.783 | 5.165 | 0.117 | 0.908 | 0.948 | 0.957 | 0.932 | 0.891 | 0.982 |
| | 100% | **0.112** | **3.361** | **4.484** | **0.102** | **0.923** | **0.953** | **0.960** | **0.953** | **0.905** | **0.982** |
| KITTI | 100% | 0.074 | 0.449 | 2.847 | 0.126 | 0.954 | 0.980 | 0.987 | 0.962 | 0.918 | 0.974 |

Table 4: Performance of 3D box estimation. The evaluation demonstrates the effectiveness of our model from each separate metrics. The models are trained and tested on the same type of dataset, either GTA or KITTI. With different amounts of training data in our GTA dataset, the results suggest that large data capacity benefits the performance of a data-hungry network.

| Range | Method | MOTA ↑ | MM ↓ | #FP ↓ | #FN ↓ |
|---|---|---|---|---|---|
| $30m$ | LiDAR [6] | 73.02 | 19.75 | **92.80** | **350.50** |
| | Ours | **77.93** | **5.29** | 104.29 | 395.33 |
| $50m$ | LiDAR [6] | 52.74 | 31.60 | **99.70** | 1308.25 |
| | Ours | **53.48** | **12.25** | 194.67 | **857.08** |
| $100m$ | LiDAR [6] | **37.98** | 32.55 | **105.40** | 2455.30 |
| | Ours | 15.59 | **19.83** | 338.54 | **1572.33** |

Table 5: Tracking performance on the validation set of Argoverse tracking benchmark [6]. Note that the LiDAR-centric baseline[6] uses LiDAR sweeps, HD maps for evaluation.

| Methods | MOTA ↑ | MOTP ↑ | MT ↑ | ML ↓ | FP ↓ | FN ↓ |
|---|---|---|---|---|---|---|
| Ours | **84.52** | 85.64 | **73.38** | **2.77** | 705 | **4242** |
| BeyondPixels† [38] | 84.24 | **85.73** | 73.23 | **2.77** | 705 | 4247 |
| PMBM† [37] | 80.39 | 81.26 | 62.77 | 6.15 | 1007 | 5616 |
| MDP [43] | 76.59 | 82.10 | 52.15 | 13.38 | 606 | 7315 |

Table 6: Tracking performance on the testing set of KITTI tracking benchmark. Only published methods are reported. † sign means 3D information is used.

comparison of these two methods on KITTI dataset. The result indicates the correct 3D projections provides higher tracking capacity for motion module to associate candidates and reduces the ID switch (IDS) significantly.

**Amount of Data Matters.** We train the depth estimation module with 1%, 10%, 30% and 100% training data. The results show how we can benefit from more data in Table 4, where there is a consistent trend of performance improvement as the amount of data increases. The trend of our results with a different amount of training data indicates that large-scale 3D annotation is helpful, especially with accurate ground truth of far and small objects.

**Real-world Evaluation.** Besides evaluating on synthetic data, we resort to Argoverse [6] and KITTI [13] tracking benchmarks to compare our model abilities. For Argoverse, we use Faster RCNN detection results of mmdetection [7] implementation pre-trained on COCO [23] dataset. Major results are listed in Table 5 and Table 6. For a full evaluation explanation, please refer to the supplementary material. Our monocular 3D tracking method outperforms all the published methods on KITTI and beats LiDAR-centric baseline methods on $50m$ and $30m$ ranges of the Argoverse tracking validation set upon the time of submission.

It is worth noting that the baseline methods on Argoverse tracking benchmark leveraging HD maps, locations, and LiDAR points for 3D detection, in addition to images. Our monocular 3D tracking approach can reach competitive results with image stream only. It is interesting to see that the 3D tracking accuracy based on images drops much faster with increasing range threshold than LiDAR-based method. This is probably due to different error characteristics of the two measurement types. The farther objects occupy smaller number of pixels, leading to bigger measurement variance on the images. However, the distance measurement errors of LiDAR hardly change for faraway objects. At the same time, the image-based method can estimate the 3D positions and associations of nearby vehicles accurately. The comparison reveals the potential research directions to combine imaging and LiDAR signals.

## 6. Conclusion

In this paper, we learn 3D vehicle dynamics from monocular videos. We propose a novel framework, combining spatial visual feature learning and global 3D state estimation, to track moving vehicles in a 3D world. Our pipeline consists of a single-frame monocular 3D object inference model and motion LSTM for inter-frame object association and updating. In data association, we introduced occlusion-aware association to solve inter-object occlusion problem. In tracklet matching, depth ordering filters out distant candidates from a target. The LSTM motion estimator updates the velocity of each object independent of camera movement. Both qualitative and quantitative results indicate that our model takes advantage of 3D estimation leveraging our collection of dynamic 3D trajectories.

## 7. Acknowledgements

The authors gratefully acknowledge the support of Berkeley AI Research, Berkeley DeepDrive and MOST-107 2634-F-007-007, MOST Joint Research Center for AI Technology and All Vista Healthcare.

## References


[1] Boris Babenko, Ming-Hsuan Yang, and Serge Belongie. Visual tracking with online multiple instance learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009. 2

[2] Keni Bernardin and Rainer Stiefelhagen. Evaluating multiple object tracking performance: the clear mot metrics. *EURASIP Journal on Image and Video Processing (JIVP)*, 2008. 6

[3] Luca Bertinetto, Jack Valmadre, Joao F Henriques, Andrea Vedaldi, and Philip HS Torr. Fully-convolutional siamese networks for object tracking. In *European Conference on Computer Vision (ECCV)*, 2016. 2

[4] David S Bolme, J Ross Beveridge, Bruce A Draper, and Yui Man Lui. Visual object tracking using adaptive correlation filters. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010. 2

[5] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. *ArXiv:1903.11027*, 2019. 2

[6] Ming-Fang Chang, John Lambert, Patsorn Sangkloy, Jagjeet Singh, Slawomir Bak, Andrew Hartnett, De Wang, Peter Carr, Simon Lucey, Deva Ramanan, and James Hays. Argoverse: 3d tracking and forecasting with rich maps. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 2, 6, 8

[7] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, Zheng Zhang, Dazhi Cheng, Chenchen Zhu, Tianheng Cheng, Qijie Zhao, Buyu Li, Xin Lu, Rui Zhu, Yue Wu, Jifeng Dai, Jingdong Wang, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. MMDetection: Open mmlab detection toolbox and benchmark. *ArXiv:1906.07155*, 2019. 8

[8] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2

[9] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. Carla: An open urban driving simulator. In *Conference on Robot Learning (CoRL)*, 2017. 2

[10] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In *Advances in Neural Information Processing Systems (NIPS)*, 2014. 6

[11] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Detect to track and track to detect. In *IEEE International Conference on Computer Vision (ICCV)*, 2017. 2

[12] Adrien Gaidon, Qiao Wang, Yohann Cabon, and Eleonora Vig. Virtual worlds as proxy for multi-object tracking analysis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2

[13] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. 2, 6, 8

[14] Ross Girshick. Fast r-cnn. In *IEEE International Conference on Computer Vision (ICCV)*, 2015. 2

[15] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 2

[16] Sam Hare, Stuart Golodetz, Amir Saffari, Vibhav Vineet, Ming-Ming Cheng, Stephen L Hicks, and Philip HS Torr. Struck: Structured output tracking with kernels. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2016. 2

[17] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *IEEE International Conference on Computer Vision (ICCV)*, 2017. 4, 6

[18] David Held, Sebastian Thrun, and Silvio Savarese. Learning to track at 100 fps with deep regression networks. In *European Conference on Computer Vision (ECCV)*, 2016. 2

[19] Zdenek Kalal, Krystian Mikolajczyk, and Jiri Matas. Tracking-learning-detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2012. 2

[20] Philipp Krähenbühl. Free supervision from video games. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2, 6

[21] Matej Kristan, Jiri Matas, Ales Leonardis, Michael Felsberg, Luka Cehovin, Gustavo Fernández, Tomas Vojir, Gustav Hager, Georg Nebehay, and Roman Pflugfelder. The visual object tracking vot2015 challenge results. In *IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, 2015. 2

[22] Peiliang Li, Tong Qin, and andShaojie Shen. Stereo vision-based semantic 3d object and ego-motion tracking for autonomous driving. In *European Conference on Computer Vision (ECCV)*, September 2018. 2

[23] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision (ECCV)*, pages 740–755. Springer, 2014. 8

[24] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European Conference on Computer Vision (ECCV)*, 2016. 2

[25] Will Maddern, Geoffrey Pascoe, Chris Linegar, and Paul Newman. 1 year, 1000 km: The oxford robotcar dataset. *International Journal of Robotics Research (IJRR)*, 2017. 2

[26] Anton Milan, Laura Leal-Taixé, Ian Reid, Stefan Roth, and Konrad Schindler. Mot16: A benchmark for multi-object tracking. *ArXiv:1603.00831*, 2016. 2

[27] Arsalan Mousavian, Dragomir Anguelov, John Flynn, and Jana Košecká. 3d bounding box estimation using deep learning and geometry. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 4, 7

[28] Aljosa Osep, Wolfgang Mehner, Markus Mathias, and Bastian Leibe. Combined image- and world-space tracking in traffic scenes. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2017. 2

[29] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2

[30] Joseph Redmon and Ali Farhadi. Yolo9000: Better, faster, stronger. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2

[31] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2015. 2, 3

[32] Stephan R Richter, Zeeshan Hayder, and Vladlen Koltun. Playing for benchmarks. In *IEEE International Conference on Computer Vision (ICCV)*, 2017. 2, 6

[33] Stephan R Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: Ground truth from computer games. In *European Conference on Computer Vision (ECCV)*, 2016. 2

[34] German Ros, Laura Sellart, Joanna Materzynska, David Vazquez, and Antonio M Lopez. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2

[35] Amir Sadeghian, Alexandre Alahi, and Silvio Savarese. Tracking the untrackable: Learning to track multiple cues with long-term dependencies. In *IEEE International Conference on Computer Vision (ICCV)*, pages 300–311, Oct 2017. 5

[36] Samuele Salti, Andrea Cavallaro, and Luigi Di Stefano. Adaptive appearance modeling for video tracking: Survey and evaluation. *IEEE Transactions on Image Processing (TIP)*, 2012. 2

[37] Samuel Scheidegger, Joachim Benjaminsson, Emil Rosenberg, Amrit Krishnan, and Karl Granstrom. Mono-camera 3d multi-object tracking using deep learning detections and pmbm filtering. In *IEEE Intelligent Vehicles Symposium (IV)*, June 2018. 2, 8

[38] Sarthak Sharma, Junaid Ahmed Ansari, J. Krishna Murthy, and K. Madhava Krishna. Beyond pixels: Leveraging geometry and shape cues for online multi-object tracking. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2018. 2, 8

[39] Arnold WM Smeulders, Dung M Chu, Rita Cucchiara, Simone Calderara, Afshin Dehghan, and Mubarak Shah. Visual tracking: An experimental survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2014. 2

[40] Ran Tao, Efstratios Gavves, and Arnold WM Smeulders. Siamese instance search for tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2

[41] Longyin Wen, Dawei Du, Zhaowei Cai, Zhen Lei, Ming-Ching Chang, Honggang Qi, Jongwoo Lim, Ming-Hsuan Yang, and Siwei Lyu. Ua-detrac: A new benchmark and protocol for multi-object detection and tracking. *ArXiv:1511.04136*, 2015. 2

[42] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. In *IEEE International Conference on Image Processing (ICIP)*, 2017. 5

[43] Yu Xiang, Alexandre Alahi, and Silvio Savarese. Learning to track: Online multi-object tracking by decision making. In *IEEE International Conference on Computer Vision (ICCV)*, pages 4705–4713, 2015. 2, 5, 8

[44] Alper Yilmaz, Omar Javed, and Mubarak Shah. Object tracking: A survey. *ACM computing surveys (CSUR)*, 2006. 2

[45] Fengwei Yu, Wenbo Li, Quanquan Li, Yu Liu, Xiaohua Shi, and Junjie Yan. Poi: multiple object tracking with high performance detection and appearance feature. In *European Conference on Computer Vision (ECCV)*, 2016. 4

[46] Fisher Yu, Dequan Wang, Evan Shelhamer, and Trevor Darrell. Deep layer aggregation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 4

[47] Fisher Yu, Wenqi Xian, Yingying Chen, Fangchen Liu, Mike Liao, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving video database with scalable annotation tooling. *ArXiv:1805.04687*, 2018. 2, 6