CAROM - Vehicle Localization and Traffic Scene Reconstruction from Monocular Cameras on Road Infrastructures

Duo Lu[1], Varun C Jammula[1], Steven Como[1], Jeffrey Wishart[2], Yan Chen[1], Yezhou Yang[1]

***Abstract*— Traffic monitoring cameras are powerful tools for traffic management and essential components of intelligent road infrastructure systems. In this paper, we present a vehicle localization and traffic scene reconstruction framework using these cameras, dubbed as CAROM, *i.e.*, "CARs On the Map". CAROM processes traffic monitoring videos and converts them to anonymous data structures of vehicle type, 3D shape, position, and velocity for traffic scene reconstruction and replay. Through collaborating with a local department of transportation in the United States, we constructed a benchmarking dataset containing GPS data, roadside camera videos, and drone videos to validate the vehicle tracking results. On average, the localization error is approximately 0.8 m and 1.7 m within the range of 50 m and 120 m from the cameras, respectively.**

I. Introduction

Traffic monitoring cameras and smart roadside units with vision-based sensors are becoming increasingly popular for traffic management purposes. Local Departments of Transportation (DOTs) use the videos to investigate driving safety, study traffic congestion, and sometimes issue tickets for rule violations. This type of equipment is also an essential component of the intelligent road infrastructure system for the "automated vehicles" in the future. However, there are three unsolved problems related to these cameras. First, transmitting and archiving the videos cost a significant amount of network bandwidth and storage space. Second, these videos are unfriendly to index, search, and automated analysis since they contain mainly unstructured information. Especially, it is difficult to obtain 3D states of the vehicles from 2D images. For example, local DOTs often require traffic management officers to monitor and interpret the videos to evaluate driving safety based on safety-critical events such as accidents. These events do not happen very frequently. The analysis can also involve subjective bias, e.g., errors in determining the velocity of vehicles or distance between vehicles due to the restriction of camera perspective angles. Third, privacy concerns haunt the civil usage of these videos, which restricts non-authority organizations to access them. For example, it is usually not acceptable for a local DOT to send the raw videos to third-party companies or scholars for data analysis unless they are contracted by the DOT. Also, insurance companies cannot access them for improving the process of traffic incident claims.

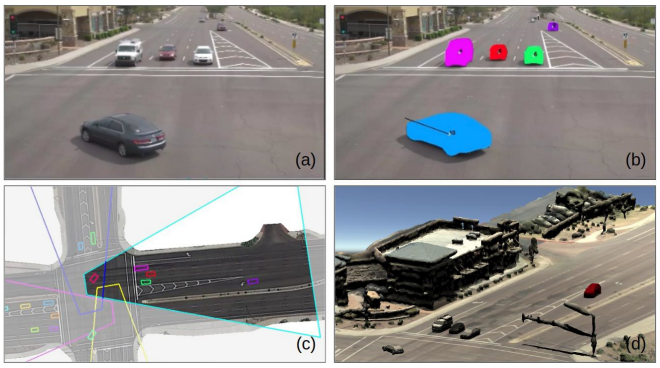

Fig. 1: An overview of CAROM: (a) original traffic monitoring video, (b) detected vehicles, (c) replay on a 2D map, (d) replay on a 3D map.

Automated vehicles (AVs) can also benefit from these cameras. For example, it is a tough question for both the DOTs and the manufacturers to answer how safe are the AVs currently testing on the road. The traffic monitoring cameras are commonly mounted on road infrastructures with the advantage of covering a large area. Hence, they can be used to objectively assess the operational safety by calculating a set of safety metrics [1] directly from vehicle movements captured on the videos. Meanwhile, the information of the surrounding traffic scene obtained by these cameras can complement the perception of AVs because the in-vehicle sensors can only reach places in the line-of-sight.

To address these issues, we propose CAROM, a framework that can extract 3D information from the videos, generate a series of structured data records of vehicle states, and reconstruct traffic scenes on a 2D map or a 3D map, as shown in Fig 1. This work is part of research being conducted by the Institute of Automated Mobility (IAM) [2] to develop an operational safety assessment methodology and an intelligent automated infrastructure for vehicles. CAROM facilitates a series of applications including road safety evaluation, roadside information services for AVs, traffic data archiving, sharing, and further automated analysis. The generated data records can be saved in a database or sent over the network with significantly less storage and bandwidth cost than the raw videos. Moreover, a reconstructed traffic scene can be replayed to offer an objective vision of the traffic situations in a bird's-eye view to an interested organization besides the video owner. Last but not the least, the generated data could be easily anonymized by removing any Personally Identifiable Information (PII). In summary, our contributions are as follows:

[1]D. Lu, V. Jammula, S. Como, Y. Chen and Y. Yang are with Arizona State University, Tempe, AZ, USA. {duolu, vjammula, scomo, yanchen, yz.yang}@asu.edu

[2] J. Wishart is with Exponent, Tempe, AZ, USA. jwishart@exponent.com

This research is sponsored by the Institute of Automated Mobility, Arizona, USA. We thank Maricopa County DOT, Niraj Vasant Altekar, Larry Head, Alex Cardona, Don Bruyere, Maria Elli, Jack Weast, Greg Leeming, and Marisa Paula Walker for their help.

1) We constructed a vehicle tracking, localization, and velocity measurement pipeline using videos taken by monocular road traffic monitoring cameras.
2) We built a reconstruction system for vehicle shapes and traffic scenes using the tracking results. Additionally, we created two visualizers to replay the reconstructed traffic scene on both 2D and 3D maps.
3) We evaluated the vehicle localization and velocity measurement performance using both differential GPS and drone videos, which shows promising results.

II. Related Work

With the advancement of effective neural network object detectors [3][4], tracking algorithms [5], and large scale datasets [6][7], current research work has obtained great achievements in video based road traffic analysis in the past decade [8][9][10][11][12][13]. Commercial video analysis software platforms as well as roadside smart cameras are also emerging [14][15]. However, localization of vehicles, speed measurement, and reconstruction of traffic scenes in 3D space are still challenging due to two core problems. First, accurate calibration of the cameras is necessary to convert 2D pixels to 3D locations. This can be done manually using labeled point correspondences or automatically using vanishing points calculated from geometric primitives [16][17][18] and objects with known shapes [19]. Typically, the automated calibration algorithm also sets up a 3D world reference frame (not related to any predefined map). Second, in addition to accurate vehicle detection and tracking on the 2D images, robust estimation of vehicle 3D pose and vehicle dimension is required. The 3D representation of a vehicle can be a point with an orientation vector [20], a 3D bounding box [16][21][22], a few key points [23], a wire-frame model [24][25], or a parametric 3D shape model [26]. The location and speed of a vehicle are typically determined from three pieces of information: (1) the 2D locations on the images, (2) the 3D poses of the vehicle, and (3) the transformation between image coordinates and the ground coordinates obtained from the camera calibration results. Usually, the vehicle states are also estimated jointly through a filtering process by considering the vehicle kinematics or dynamics [27][28]. The vehicle shape can be reconstructed using stereo cameras [29] or monocular cameras [25][30][31] through a sequence of algorithms for depth estimation, model fitting, and shape optimization. Our paper is based on the existing works for several individual computer vision tasks and we integrated them to a unified framework that extracts the location, speed, and vehicle shape in the 3D space. Further, our tracking results allow the vehicle movements to be replayed on a 2D map or a 3D map so as to support traffic analysis tasks.

The use of simulators with a single vehicle or a collection of vehicles have been studied intensively to visualize vehicle motion, study vehicle dynamics, understand traffic patterns, and train driving behaviors of AVs [32][33]. Unlike these works, we desire to "re-simulate" the traffic scenes using the reconstruction results from the videos.

III. The CAROM Framework Architecture

The CAROM framework consists of three subsystems, as illustrated in Fig 2. The first one is the tracking system, which runs a pipeline to generate data structures of vehicle states from videos. This pipeline contains an offline calibration stage (detailed in Section III.A) and a few online video processing stages, including vehicle detection, tracking, localization, type recognition, and 3D state estimation (detailed in Section III.B). The generated data structures can be stored in files or a database for future usage, such as road safety assessment. The second one is the reconstruction system for vehicle shapes (detailed in Section III.C) and the map (detailed in Section III.D). The third subsystem is the replay engine that animates the traffic scene on the reconstructed map using the tracking results (detailed in Section III.E).

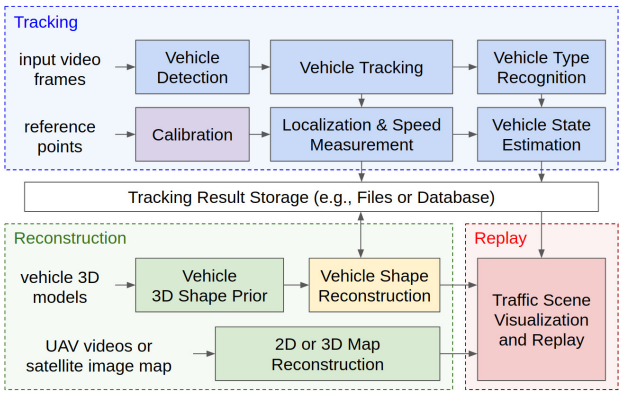

Fig. 2: The CAROM Framework Architecture

A. Camera and Map Calibration

CAROM uses a pinhole camera model and assumes the camera distortion is negligible, as illustrated in Fig. 3. The ground is modeled either as a flat surface corresponding to a 2D satellite image map or a 3D surface with a high-resolution 3D mesh map, as shown in Fig. 4. There are three reference frames: (1) the camera frame in image pixel coordinates, (2) the world frame in metric coordinates, and (3) the map frame in map coordinates. For a 2D map, the origin is the top-left corner, the axes follow east-south directions, and the unit is a map pixel (as in Fig. 3). For a 3D map, the origin can be any point on the ground surface, the axes follow east-north-up directions, and the coordinates use the metric unit (as in Fig. 4). The calibration procedure constructs two sets of parameters: (1) a camera projection matrix from the world frame to the camera frame, (2) a transformation between the map frame and the world frame. Since the traffic monitoring cameras do not move, we only need to run the calibration procedure once for each camera in the following steps. First, we label a set of at least six point correspondences on the map and the image, typically using the lane markers and features on the ground. Second, we create the world frame and compute the transformation between the world frame to the map frame. Usually, the XOY plane of the world frame is the ground plane in the 2D map and the x-axis follows the traffic moving direction. Third, we transform the

labeled points on the map to the world frame and compute the camera projection matrix from the point correspondences [34]. Optionally, the calibration of the camera can be automated [16], but the transformation between the world frame and the map still needs to be determined using labeled point correspondences. The transformation from any image coordinates to the world frame on the ground is crucial for vehicle localization, and we denoted it as T. If a 2D map is used, T is the planar homography between the camera frame and the ground plane, which is derived from the camera projection matrix. If the 3D map is used, we back-project each pixel on the image to the 3D ground surface to obtain its corresponding point in the world frame. Then we construct T as a look-up-table. Additionally, we also compute the horizon line on the image from the camera projection matrix.

B. Online Vehicle Tracking Pipeline

The tracking pipeline consists of a set of online algorithms that independently processes every image in a video. It uses information from the previous images for vehicle speed measurement. With enough processing resources, it may be able to run in real-time. It has the following stages.

(1) Vehicle Detection: For each video image, the system runs an object detection and instance segmentation network. In our implementation, We fine-tuned a Mask RCNN [4] on a custom dataset created from traffic monitoring videos for this step. The quality of the masks is crucial since the later localization stage relies on the contour of the mask.

(2) Vehicle Tracking: For each detected object instance, its 2D bounding box on the current image is enlarged four times as a region-of-interest (ROI). The sparse optical flow vectors [35] from the previous image to the current image are calculated within this ROI and on the masks. The detected instances on the two images are associated in linked lists using these vectors and the mask overlapping percentages.

(3) Vehicle Type Recognition: For each detected object instance, the system crops a square patch from the image just large enough to contain its 2D bounding box, resizes the cropped patch and runs a classifier to predict its type. The following types are used: {pedestrian, two-wheelers, bus, mini-truck, semi-truck, pickup-truck, convertible, coupe, sedan, all-terrain vehicle, minivan, van, SUV, trailer}. In our implementation, we trained a ResNet-18 [36] on a custom dataset for this step. The decoupling of detector and vehicle type classifier is intentional, which makes both neural networks easier to train. Since this recognizer can learn a different set of features dedicated to its task regardless of the detector, it may also perform better. Moreover, we plan to build a fine-grained vehicle make and model classifier to replace this vehicle type recognizer in the future.

(4) Vehicle Localization: The system runs RANSAC [37] on the computed optical flow vectors obtained in the previous vehicle tracking stage to select those vectors that meet at the same vanishing point on the horizon line (which is computed from the camera calibration results), as shown in Fig. 5. Because a vehicle rarely moves backward on the road, the vehicle heading is determined by the line from the center of its 2D bounding box to this vanishing point. The vehicle has its XYZ coordinate reference frame where the x-axis points to the vehicle's forward, and the y-axis points to its left. We assume that the vehicle is always on the ground, *i.e.*, its z-axis is pointing up relative to the ground surface. With the center of its 2D bounding box as its temporary location on the image, the transformation T, and its heading, the system computes the other two vanishing points corresponding to the y-axis and z-axis of the vehicle. Finally, using all three vanishing points, the 3D bounding box of a vehicle is computed from the contour of its segmentation mask using the tangent line method [16] (illustrated in Fig. 5). We made several improvements in implementation details to handle a few particular viewing angles not considered in [16], and we also made adjustments on the computed 3D bounding box using empirical results to accommodate vehicles without "boxy" shapes. Besides, we use the recognized vehicle type and prior knowledge of the vehicle dimensions for different vehicle types to adjust the 3D bounding box dimensions. Additionally, the 3D bounding box is not calculated if certain occlusion conditions are detected using the 2D bounding box overlap and the size of the mask. Finally, the center of this 3D bounding box's bottom surface is the vehicle's location on the image. Again, with the transformation T, the location of the vehicle in the world frame is obtained.

The heading calculation may fail in a few cases: (a) when the vehicle stops, (b) when the vehicle is far away with only small motion on the image, or (c) when the RANSAC fails. In these cases, the heading is inferred from the accumulated motion on several previous video images by assuming the vehicle travels in a straight line within a short amount of time. Moreover, we use a neural network similar to [20] to predict the heading angle of a vehicle on the image as a backup. It is trained on a dataset of images patches with correct heading calculated by the optical flow based method. However, this neural network method is generally slower, less accurate and less robust than the optical flow method.

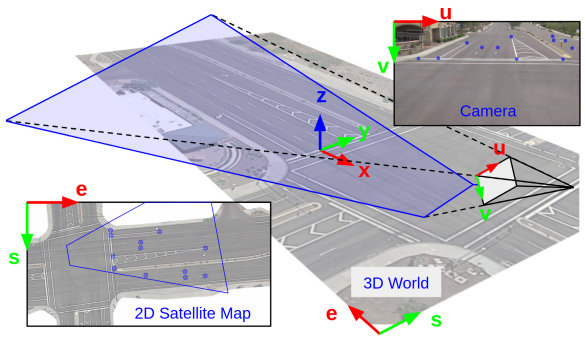

Fig. 3: An illustration of reference frames and point correspondences.

Fig. 4: An example of the 3D map and camera coverage.

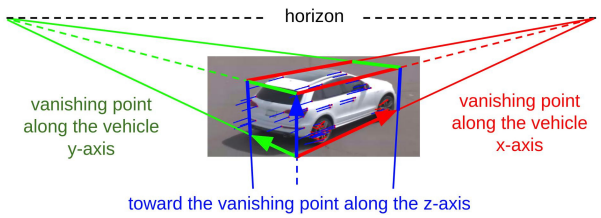

Fig. 5: An illustration of 3D bounding box. The optical flow vector inliers are shown as the thin blue lines (on the vehicle body) and outliers are shown as the thin red lines (on the wheels).

(5) Vehicle Velocity Measurement: The system first averages the length of the inlier vectors of the RANSAC results obtained in the previous step and use this length as the distance of vehicle movement from the current image and the previous image. Next, it also obtains the corresponding distance in the world frame using the vehicle's location, heading, and the transformation T. After that, the system uses the linked list of associated instances to aggregate these distances calculated from previous image pairs in sequence until the total distance exceeds a threshold or up to a certain amount of steps (5 m or 30 steps in our implementation). Finally, the velocity is calculated from this aggregated distance, the number of frame pairs, and the frame interval time. The direction of the velocity is the same as the heading.

(6) Vehicle State Estimation: Given the location and the velocity of a vehicle, the system runs a Kalman filter with states $\mathbf{x} = (x, y, z, \dot{x}, \dot{y}, \dot{z})$ and linear 6DOF rigid body kinematics to estimate the vehicle states in the 3D world. The process and observation noise covariance matrices are empirically determined. For a 2D map, z and $\dot{z}$ are always zero. The prediction step of the state estimation will keep running for a few iterations when the detection of this vehicle fails. Once it is detected again, a detected instance can be re-associated to this vehicle. The heading and 3D bounding box dimension is also smoothed using a running average on previous video images. Finally, a record of the location, velocity, heading, 3D bounding box points, and the vehicle type is created as the output of this pipeline.

C. Vehicle Shape Reconstruction

We applied the tracking pipeline on the traffic monitoring videos obtained from four cameras pointing to the four directions of an intersection, as shown in Fig. 6 (right). These videos allow us to observe the same vehicle traveling through the intersection from multiple viewing angles. Given a sequence of vehicle locations, 3D bounding boxes, and segmentation masks from multiple images, we compute the vehicle's visual hull using the shape-from-silhouette method [38]. Specifically, we first initialize a rectangular cuboid of voxels using the 3D bounding box. Then we carve those voxels that cannot be projected onto any mask on any views. After that, we further process the remaining voxels using the symmetry property along the vehicle's x-axis. The voxels can be converted to a mesh using the marching cubes algorithm [39] for visualization, as in Fig. 6 (left). The voxels can also be converted to a 2D histogram by ignoring the details on the bottom side. Specifically, for each histogram bin at (x, y)

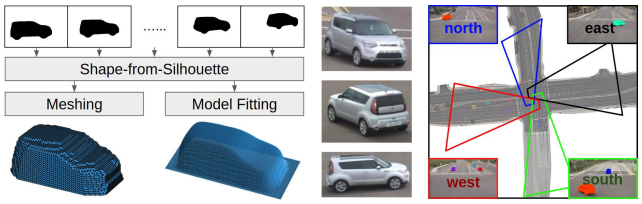

Fig. 6: The vehicle shape reconstruction pipeline with the reconstructed 3D shape of an example vehicle (left), three images of the vehicle (middle), and the intersection with four cameras providing the videos (right).

in the vehicle's own XYZ coordinate frame, the histogram value is the maximum of the z coordinates of all remaining voxels with this (x, y) coordinates. Similarly, a 2D histogram can be converted back to voxels. We further resample the 2D histogram to a fixed size of n-by-m bins using bilinear interpolation. The result histogram is denoted as a n-by-m matrix H, or flattened as a n*m dimensional vector $\mathbf{h}$. In our implementation, $n = m = 50$.

Our objective is to reconstruct the vehicle shape and representing it in a fixed-sized data structure. Here H can be a candidate. However, it usually differs from the actual vehicle shape due to the limited view angles in the voxel carving process and errors in the localization results. To solve this problem, we construct a shape prior model from 80 different 3D CAD vehicle models and fit the model to the reconstructed histogram H by the following procedures.

Step (1): For each 3D CAD model, we converted it to a histogram using an algorithm similar to the one that converts voxels to a histogram. The generated histogram was resampled to n-by-m, and flattened to a n*m vector (denoted as the model vector $\{\mathbf{u}_i\}$, $1 \le i \le 80$).

Step (2): With all 80 model vectors, we run Principal Component Analysis (PCA) to reduce their dimension from n*m to 20. After this step we obtained 20 principle component vectors (denoted as a n*m-by-20 matrix S). The vector set $\{\mathbf{u}_i\}$ and matrix S are called the **vehicle shape prior**, similar to the shape prior models in shape analysis and multiple view reconstruction [40][29][30].

Step (3): We projected the reconstructed histogram $\mathbf{h}$ to the column space spanned by S by solving the following least-square problem:

$$\arg\min_{\mathbf{v}} ||\mathbf{h} - S\mathbf{v}|| + \lambda ||\mathbf{v} - \mathbf{t}||.$$

Here the last term is a regularizer, $\mathbf{t}$ is a template vector for the type of the reconstructed vehicle, and it is computed by averaging the subset of $\{\mathbf{u}_i\}$ with the same type. For example, if $\mathbf{h}$ is reconstructed from a vehicle that is recognized as a "sedan", $\mathbf{t}$ is the template vector of "sedan", which is computed by averaging of those $\mathbf{u}_i$ derived from 3D models of sedans. Moreover, $\mathbf{t}$ is also used to represent those vehicles whose shapes cannot be reconstructed due to occlusion.

Finally, the vector $\mathbf{v}$ is the output of this pipeline as the shape representation of the reconstructed vehicle. Given $\mathbf{v}$ and S, an approximated histogram representation of the vehicle shape can be recovered by $\hat{\mathbf{h}} = S\mathbf{v}$. This histogram can be further converted to voxels or a mesh. The texture of the vehicle 3D model is not reconstructed for anonymity.

Videos	MOTA	MODA	MME	FP	FN	#Objects	#Images	#Vehicles	#IDE	#TO	#PO	MT	ML	Resolution
Track 1A	96.2%	98.1%	220	20	3,802	95,227	17,891	286	22	7	112	271	5	720p
Track 1B	90.9%	92.4%	1,218	330	5,670	79,210	17,912	225	40	10	109	207	6	720p
Track 2	95.5%	96.3%	65	0	496	12,346	25,458	80	1	0	2	75	2	1080p

TABLE I: Tracking results.

D. Map Reconstruction

We constructed the 2D map using a satellite image at the place where the camera is mounted. Many online map services (such as Google Maps) offer satellite images. The rows and columns of the image are usually already aligned to the east and the south. We also calculate the scale factor between the map pixel and the metric unit using two points with known actual distance in the metric unit (which can be measured using the online map service tools or in the world).

For the 3D map, we flew a survey-grade drone (DJI Phantom Pro with RTK) on the site, ran a 3D reconstruction software (Pix4D mapper) to obtain a point cloud from the drone images, and then processed the point cloud to a 3D mesh map. We also calibrated this mesh map to align its axes to east-north-up and recover the actual scale.

Additionally, we chose a reference point for both types of maps and obtained its longitude, latitude, and height above the geodesic ellipsoid using the online map service or a hand-held GPS receiver device on site. Then, we set up the transformation between the map reference frame to the WGS84 reference frame so that we can compare our localization results with GPS measurements.

E. Traffic Scene Visualization and Replay

To replay a traffic scene captured by the cameras, we built two visualizers, one using the 2D map and the other using the 3D map, as shown in Fig. 1. Here, a traffic scene is defined as the collection of the road environment (*i.e.*, the map) and vehicles captured by a specific camera within a certain period (*i.e.*, the tracking results). Both visualizers transform the vehicle states to the map frame using the calibration results and animate the vehicle movement. We use a template 3D mesh model for each vehicle type or the reconstructed vehicle models for the 3D animation. The size of the 3D model is scaled to fit the vehicle 3D bounding box. Besides, during the replay, the user can modify the speed of one specific vehicle, and the visualizer can "re-simulate" this vehicle from the modified states following the recorded trajectory while keeping replaying other vehicles.

IV. Empirical Evaluation

We obtained traffic monitoring videos from two sites. The first one is an intersection with four cameras pointing to its four directions (the same intersection in [1]), which is the one shown in Fig. 6 in section III.C. The second site is a local road segment with one camera, shown in Fig. 7.

First, we evaluate the vehicle type recognition performance. The recognizer is trained with 10,200 images and tested with 883 images. All images are cropped from the videos recorded by the four cameras at the first site. The overall accuracy is 84%. The majority of the wrong predictions are among the following type pairs: (SUV, sedan), (SUV, minivan), (sedan, coupe). Prediction errors are more frequent when only the frontal side or the rear side of the vehicle is visible, *i.e.*, when the vehicle is driving directly towards the camera or away from the camera.

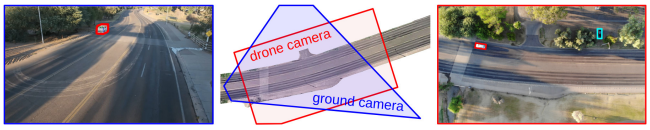

Fig. 7: Example images taken by the ground camera on the road infrastructure (left) and the drone at a height of 80 meters (right) at the second site. The coverage of both cameras are shown on the map (middle).

Video	L-Diff (m)	V-Diff (m/s)	#Vehicles (w/ Ref)	Coverage (m)	Ref Device
Track 1A	2.05	1.01	1	25 ∼120	GPS
Track 1B	1.57	0.69	1	25 ∼120	GPS
Track 2	1.68	1.47	69	15 ∼110	Drone

TABLE II: Localization and speed measurement results.

Second, we evaluate the tracking performance on the 2D images with two video tracks from the first site (the eastbound and the southbound) and one video track from the second site. The results are shown in TABLE I, mostly following the metrics in [41]. Here "#Objects", "#Images" and "#Vehicles" mean the total number of objects, images, and vehicles in the video track. "#IDE" means the number of vehicles that have tracking ID errors. "#TO" means "total occlusions", which is the number of vehicles that are occluded by other vehicles in at least one image such that more than 80% of the vehicle is not visible. Generally, the detector fails to detect them and the tracker needs to re-associate it later. If a vehicle is partially occluded but still detected, it is counted in "#PO", which means "partial occlusions". Typically, the traffic monitoring cameras are mounted at strategically chosen places to minimize occlusion. "#MT" is the number vehicles that are tracked more than 80% of the life span (*i.e.*, "mostly tracked"). "#ML" is the number vehicles that are tracked less than 20% of the life span (*i.e.*, "mostly lost"). Our system is able to track most detected vehicles that are not completely occluded.

Third, we quantitatively evaluate the vehicle localization and velocity measurement performance in the 3D world frame using two different types of references. For the first site, we drove a vehicle with a differential GPS receiver through the intersection. For the second site, we flew a drone and capture videos from 80 meters above the road. We processed the drone videos to obtain the vehicle location and velocity using a method similar to [42]. We also compared the location measurements between the GPS and the drone using another drone video track that captures the movement of a vehicle with a differential GPS receiver. The location and velocity measurements accuracy of both types of reference devices are within 1 m and 1 m/s respectively. Our results are shown in TABLE II, where "L-Diff" and "V-Diff" mean

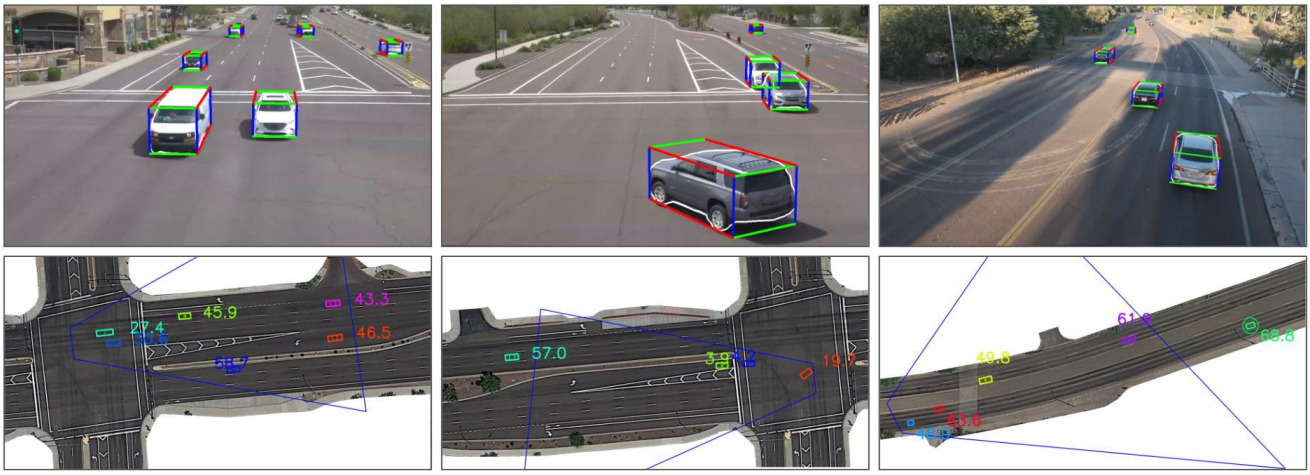

Fig. 8: Examples of reconstructed traffic scenes. The first row shows the original video with vehicle 3D bounding boxes. The second row shows on the map with the vehicle location with uncertainty range (the rectangles and the circles on them) and speed in km/h (the numbers adjacent to the rectangles).

Fig. 9: Examples of reconstructed vehicles with voxel representations and histogram shapes.

differences in location and velocity between our results and the reference measurements. Only the vehicles that are correctly tracked and those with corresponding reference measurements are evaluated, as shown in the "#Vehicles" columns. The "Coverage" column shows the minimum distance and the maximum distance between the measured vehicles and the camera. Note that "L-Diff" actually varies with the distance between the vehicle and the camera, and typically it is smaller when the vehicle is close to the camera. For example, the average value of "L-Diff" is 0.79 m within 50 m to the camera in track 2, which is less than the average "L-Diff" value in the whole range. Also, "L-Diff" is not the same in the longitudinal direction (*i.e.*, the camera pointing direction) and the lateral direction (*i.e.*, perpendicular to the camera pointing direction). For example, in track 2, the average lateral location difference is just 0.22 m but the average longitudinal location difference is 1.52 m. At certain view angles, e.g., when the vehicle is driving directly towards the camera, the 3D bounding box is inaccurate. Moreover, the road surface is not perfectly flat, which can cause errors in the conversion of image coordinates to the world coordinates when the 2D map is used. Converting world coordinates to GPS coordinates may also introduce small errors.

Fourth, we show the qualitative results in Fig. 8, Fig. 9, and the supplemental videos[1]. We also obtained positive feedback from Maricopa County DOT and the Institute of Automated Mobility [2] in Arizona, which confirmed that these results are useful to some extent for tasks such as traffic counting and driving safety analysis.

[1]https://github.com/duolu/CAROM

At last, we discuss the limitations of the current system and possible future improvements. Currently, both the 3D bounding box calculation and the shape-from-silhouette algorithm require a complete segmentation mask. Inaccurate results are generated for vehicles with partial occlusion. We plan to collect a large-scale dataset and train a neural network pose estimator that can work robustly under partial occlusion. We also aim to train a neural network to directly predict the vehicle shape vectors and its pose jointly from a single image using the current vehicle reconstruction results as training data, so that the 3D shape of a vehicle can be directly obtained at every frame. This will enable us to develop a model-based tracking algorithm to increase the processing speed and improve the robustness. Besides, pedestrians, cyclists, and other types of traffic participants are not reconstructed in our current implementation, and we want to calculate "bounding cylinders" for these moving objects with non-boxy shapes. Finally, we are working on calculating safety metrics [1] from our tracking results as an application.

V. Conclusions

In this paper, we present CAROM, a vehicle localization and traffic scene reconstruction framework using videos taken by monocular cameras mounted on road infrastructures, which achieves promising results for vehicle localization and velocity measurement. Still, CAROM is in its early stage with limitations in robustness and efficiency. With further development, we hope it can be deployed together with traffic monitoring cameras on the roadside infrastructure in the future, to allow jurisdictional authorities and AVs on the road gain better awareness of the traffic situation.

REFERENCES

[1] J. Wishart, S. Como, M. Elli, B. Russo, J. Weast, N. Altekar, E. James, and Y. Chen, "Driving safety performance assessment metrics for ads-equipped vehicles," *SAE International Journal of Advances and Current Practices in Mobility*, vol. 2, no. 2020-01-1206, pp. 2881–2899, 2020.

[2] "Institute of Automated Mobility," http://www.azcommerce.com/iam, accessed: 2021-03-20.

[3] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2015, pp. 91–99.

[4] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 2961–2969.

[5] N. Wojke, A. Bewley, and D. Paulus, "Simple Online and Realtime Tracking with a Deep Association Metric," in *2017 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2017, pp. 3645–3649.

[6] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision Meets Robotics: The KITTI Dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[7] M. Naphade, S. Wang, D. C. Anastasiu, Z. Tang, M.-C. Chang, X. Yang, L. Zheng, A. Sharma, R. Chellappa, and P. Chakraborty, "The 4th AI City Challenge," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2020, pp. 626–627.

[8] S. Gupte, O. Masoud, R. F. Martin, and N. P. Papanikolopoulos, "Detection and classification of vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 3, no. 1, pp. 37–47, 2002.

[9] S. Sivaraman and M. M. Trivedi, "Looking at Vehicles on the Road: A Survey of Vision-based Vehicle Detection, Tracking, and Behavior Analysis," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 4, pp. 1773–1795, 2013.

[10] Y. Liu, B. Tian, S. Chen, F. Zhu, and K. Wang, "A Survey of Vision-based Vehicle Detection and Tracking Techniques in ITS," in *Proceedings of 2013 IEEE International Conference on Vehicular Electronics and Safety*. IEEE, 2013, pp. 72–77.

[11] S. K. Kumaran, D. P. Dogra, and P. P. Roy, "Anomaly Detection in Road Traffic Using Visual Surveillance: A Survey," *arXiv preprint arXiv:1901.08292*, 2019.

[12] B. Tian, Q. Yao, Y. Gu, K. Wang, and Y. Li, "Video Processing Techniques for Traffic Flow Monitoring: A Survey," in *2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2011, pp. 1103–1108.

[13] S. R. E. Datondji, Y. Dupuis, P. Subirats, and P. Vasseur, "A Survey of Vision-based Traffic Monitoring of Road Intersections," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 10, pp. 2681–2698, 2016.

[14] "Transoft Solutions," https://www.transoftsolutions.com/.

[15] "NoTraffic AI Sensor and Road Infrastructure Management Solution," https://notraffic.tech/how-it-works/.

[16] M. Dubská, A. Herout, and J. Sochor, "Automatic Camera Calibration for Traffic Understanding," in *The British Machine Vision Conference (BMVC)*, vol. 4, no. 6, 2014, p. 8.

[17] E. R. Corral-Soto and J. H. Elder, "Automatic Single-View Calibration and Rectification from Parallel Planar Curves," in *European Conference on Computer Vision (ECCV)*. Springer, 2014, pp. 813–827.

[18] S. C. Lee and R. Nevatia, "Robust Camera Calibration Tool for Video Surveillance Camera in Urban Environment," in *CVPR 2011 Workshops*. IEEE, 2011, pp. 62–67.

[19] J. Sochor, R. Juránek, and A. Herout, "Traffic Surveillance Camera Calibration by 3D Model Bounding Box Alignment for Accurate Vehicle Speed Measurement," *Computer Vision and Image Understanding*, vol. 161, pp. 87–98, 2017.

[20] R. Juranek, A. Herout, M. Dubská, and P. Zemcik, "Real-Time Pose Estimation Piggybacked on Object Detection," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 2381–2389.

[21] J. Sochor, A. Herout, and J. Havel, "Boxcars: 3D Bounding Boxes as CNN Input for Improved Fine-Grained Vehicle Recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 3006–3015.

[22] A. Mousavian, D. Anguelov, J. Flynn, and J. Kosecka, "3D Bounding Box Estimation Using Deep Learning and Geometry," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 7074–7082.

[23] S. Zhang, C. Wang, Z. He, Q. Li, X. Lin, X. Li, J. Zhang, C. Yang, and J. Li, "Vehicle Global 6-DoF Pose Estimation under Traffic Surveillance Camera," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 159, pp. 114–128, 2020.

[24] W. Ding, S. Li, G. Zhang, X. Lei, and H. Qian, "Vehicle Pose and Shape Estimation Through Multiple Monocular Vision," in *2018 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2018, pp. 709–715.

[25] J. A. Ansari, S. Sharma, A. Majumdar, J. K. Murthy, and K. M. Krishna, "The Earth ain't Flat: Monocular Reconstruction of Vehicles on Steep and Graded Roads from a Moving Camera," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 8404–8410.

[26] M. J. Leotta and J. L. Mundy, "Vehicle Surveillance with a Generic, Adaptive, 3D Vehicle Model," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 7, pp. 1457–1469, 2010.

[27] S. Chen, "Kalman Filter for Robot Vision: a Survey," *IEEE Transactions on Industrial Electronics*, vol. 59, no. 11, pp. 4409–4420, 2011.

[28] J. Li, W. Zhan, and M. Tomizuka, "Generic Vehicle Tracking Framework Capable of Handling Occlusions based on Modified Mixture Particle Filter," in *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2018, pp. 936–942.

[29] F. Engelmann, J. Stückler, and B. Leibe, "SAMP: Shape and Motion Priors for 4D Vehicle Reconstruction," in *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2017, pp. 400–408.

[30] F. Chhaya, D. Reddy, S. Upadhyay, V. Chari, M. Z. Zia, and K. M. Krishna, "Monocular Reconstruction of Vehicles: Combining SLAM with Shape Priors," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 5758–5765.

[31] V. A. Prisacariu, A. V. Segal, and I. Reid, "Simultaneous Monocular 2D Segmentation, 3D Pose Recovery and 3D Reconstruction," in *Asian Conference on Computer Vision (ACCV*. Springer, 2012, pp. 593–606.

[32] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An Open Urban Driving Simulator," *arXiv preprint arXiv:1711.03938*, 2017.

[33] P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, and E. WieBner, "Microscopic Traffic Simulation using SUMO," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2018, pp. 2575–2582.

[34] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2003.

[35] B. D. Lucas, T. Kanade *et al.*, "An Iterative Image Registration Technique with an Application to Stereo Vision," 1981.

[36] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[37] 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*, vol. 24, no. 6, pp. 381–395, 1981.

[38] A. Laurentini, "The Visual Hull Concept for Silhouette-based Image Understanding," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 2, pp. 150–162, 1994.

[39] W. E. Lorensen and H. E. Cline, "Marching Cubes: A High Resolution 3D Surface Construction Algorithm," *ACM SIGGRAPH Computer Graphics*, vol. 21, no. 4, pp. 163–169, 1987.

[40] T. F. Cootes, C. J. Taylor, D. H. Cooper, and J. Graham, "Active Shape Models - Their Training and Application," *Computer Vision and Image Understanding*, vol. 61, no. 1, pp. 38–59, 1995.

[41] R. Stiefelhagen, K. Bernardin, R. Bowers, J. Garofolo, D. Mostefa, and P. Soundararajan, "The CLEAR 2006 evaluation," in *International Evaluation Workshop on Classification of Events, Activities and Relationships*. Springer, 2006, pp. 1–44.

[42] W. Zhan, L. Sun, D. Wang, H. Shi, A. Clausse, M. Naumann, J. Kummerle, H. Konigshof, C. Stiller, A. de La Fortelle *et al.*, "Interaction Dataset: An International, Adversarial and Cooperative Motion Dataset in Interactive Driving Scenarios with Semantic Maps," *arXiv preprint arXiv:1910.03088*, 2019.