DeepTAgent: Realtime Tracking of Dense Traffic Agents Using Heterogeneous Interaction

Rohan Chandra\textsuperscript{1}, Tanmay Randhavane\textsuperscript{2}, Uttaran Bhattacharya\textsuperscript{1}, Aniket Bera\textsuperscript{2}, and Dinesh Manocha\textsuperscript{1}

http://gamma.cs.unc.edu/HTI

Abstract—We present a realtime algorithm to track different traffic agents in dense videos. Our approach is designed for heterogeneous traffic scenarios, which consist of different agents including vehicles, bicycles, pedestrians, two-wheelers, etc., sharing the road. We present a novel heterogeneous traffic motion and interaction model (HTMI) to predict the trajectories and interaction between the agents. We combine HTMI with the tracking-by-detection paradigm and use CNNs to compute the features of traffic agents for accurate tracking reliably. We highlight the performance on a new dataset of dense traffic videos and observe 72.02% accuracy. Our approach can handle all kinds of traffic videos in realtime on a single GPU. We observe \(4X\) speedup over prior tracking algorithms and more than 7% improvement in accuracy.

I. INTRODUCTION

Tracking of traffic agents on a highway or an urban road is an important problem in autonomous driving, computer vision, intelligent transportation, and related areas. These heterogeneous traffic agents may correspond to large or small vehicles, buses, bicycles, rickshaws, pedestrians, moving carts, etc. Different agents have different shapes, move at varying speeds, and their trajectories are governed by underlying dynamics constraints. Furthermore, the traffic patterns or behaviors can vary considerably between highway traffic, urban traffic or driving in highly congested areas (e.g., in Asian cities).

Given a traffic video, the tracking problem corresponds to computing the consistency in the temporal and spatial identity of all agents in the image sequence. There is extensive work in vehicle and object tracking, and different methods have been proposed based on the use of cameras or laser rangefinders. Recent developments in autonomous driving and large-scale deployment of high-resolution cameras for surveillance has generated interest in the development of realtime tracking algorithms, especially in dense scenarios with a large number of heterogeneous agents. The complexity of tracking increases as different types of traffic agents are in close proximity and interact with each other, including vehicle-vehicle, vehicle-pedestrian, vehicle-bicycle, pedestrian-pedestrian, etc.

The traffic congestion on highways and urban roads often result in high-density traffic scenarios. The traffic density can be defined based on the number of distinct traffic agents captured in a single frame of the video or the number of agents per unit length (e.g., a kilometer) of the roadway. It is not uncommon to capture videos with tens or hundreds of traffic agents in a single frame. The high density makes it hard to track all the agents reliably over a sequence of frames.

Main Contributions: We present a novel realtime algorithm (DeepTAgent) to track traffic agents in dense videos. Our approach makes no assumptions about the agents or their motion or lighting conditions. We combine deep learning based agent detection scheme (i.e., tracking-by-detection paradigm) with a novel heterogeneous traffic motion and interaction (HTMI) model to compute the features of each agent and perform reliable tracking. The novel contributions of our work include:

1) We present “Heterogeneous Traffic Motion and Interaction Model” (HTMI), to model the motion of different traffic agents as well as the pairwise interaction between nearby agents. It is general and applicable to all traffic agents.

2) We present an algorithm to automatically compute Deep TA-Features of each agent using HTMI and CNNs. These features reduce the number of false negatives and identity switches and significantly increase the accuracy of our algorithm in dense videos.

3) We have a collected a high-resolution traffic dataset corresponding to dense videos of highway and urban traffic with varying lighting conditions and camera angles. Furthermore, these videos capture the traffic patterns and behaviors from different geographic regions of the world, including the USA, China, and India. We
have evaluated the performance of DeepTAgent and observe 72.02% tracking accuracy.

We have also compared the performance with prior tracking methods on KITTI and PETS datasets and observe an improvement of 7% in the number of successful tracks recovered, along with a speedup of >4X.

The rest of the paper is organized as follows: In Section II, we briefly survey the state-of-the-art works in traffic modeling, vehicle and pedestrian tracking. We describe our tracking algorithm, DeepTAgent, and HTMI in Section III. In Section IV, we present the tracking results on our dense traffic videos and also compare the accuracy with prior methods on other datasets.

II. RELATED WORK

In this section, we give a brief overview of prior work on object tracking and motion modeling.

A. Pedestrian and Vehicle Tracking

There is extensive work on pedestrian tracking [1], [2]. Bruce et al. [3] and Gong et al. [4] predict pedestrians' motions by estimating their destinations. Liao et al. [5] compute a Voronoi graph from the environment and predicts pedestrian’s motion along the edges. Mehran et al. [6] apply the social force model to detect people’s abnormal behaviors from videos. Pellegrini et al. [7] use an energy function to build a goal-directed short-term collision-avoidance motion model. Bera et al. [8] use reciprocal velocity obstacles and hybrid motion models to improve the accuracy. All these methods are specifically designed for tracking pedestrian movement.

Vehicle tracking has been studied in computer vision, robotics, and intelligent transportation. Some of the earlier techniques are based on using cameras [9], [10], [11] and laser range finders [12], [13]. The recent developments in autonomous driving have resulted in the development of better sensors and new methods. [14] model dynamic and geometric properties of the tracked vehicles and estimate their positions. Using a stereo rig mounted on a mobile platform. [15] present an approach to detect and track vehicles in highly dynamic environments. [16], [17] use multiple cameras for tracking all surrounding vehicles. Moras et al. [18] use an occupancy grid framework to manage different sources of uncertainty for more efficient vehicle tracking, Wojke et al. [19] use LiDAR for moving vehicle detection and tracking in unstructured environments. [20] uses a feature-based approach to track the vehicles under varying lighting conditions. Most of these methods focus on vehicle tracking and do not take into account interactions with other traffic agents like pedestrians or bicycles in dense urban environments.

B. Motion Models and Tracking

There is substantial work on tracking multiple objects and use of motion models to improve the accuracy [21], [22], [23]. [21] presents an extension to MHT [24] so that it can be used with tracking-by-detection paradigm. [23] uses the constant velocity motion model to join fragmented pedestrian tracks caused by occlusion. RVO [25] is a non-linear motion model that has been used for pedestrian tracking in dense crowd videos. Recently, an extension to RVO has been proposed to model the trajectories of heterogeneous traffic agents with kinematic constraints [26]. Other motion models have been used for pedestrian tracking, including social forces [27], LTA [28], and ATTR [29].

C. Traffic Modeling and Navigation

Prior work in transportation engineering and robotics has focused on modeling the movement of vehicles and other road agents [30]. Traffic flow can be modeled using macroscopic [31], [32] or microscopic [33], [34], [35], [36] techniques. Luo et al. [37] propose a cellular automata model to simulate the car and bicycle heterogeneous traffic on an urban road. Chow et al. [38] model dynamic traffic based on the variational formulation of kinematic waves. Recent work in autonomous driving includes modeling human interactions or actions [39]. Our approach to model heterogeneous interactions is designed for dense traffic scenarios and complimentary to these approaches.

III. DEEP TAGENT: TRACKING HETEROGENEOUS AGENTS

In this section, we present our novel tracking algorithm that combines Mask R-CNN with a novel heterogeneous interaction model. One of the major challenges with tracking heterogeneous agents in dense traffic is that these agents corresponding to cars, buses, bicycles, pedestrians, etc. have different sizes, geometric shape, maneuverability, behavior, and dynamics (Figure 3). These often lead to complex inter-agent interactions that are usually not taken into account by prior multi-object trackers. For example, a bicyclist is riding next to a car, or a passenger is getting down from a bus. Furthermore, there are high-density scenarios, and these traffic-agents are in close-proximity or almost colliding.
configurations. So we need efficient techniques to predict their motion and interactions.

**Overview:** An overview of our approach is given in Fig. 2. All the symbols and notation used in the paper is highlighted in Table I. Our tracking algorithm starts by detecting all agents in $F_t$ using Mask R-CNN as part of traffic agent detection. Mask R-CNN segments out the shape of each agent from its bounding box. These segmented representations consist of the exact shape of the agent overlaid on a white background. These representations are important for tracking accuracy. Furthermore, they are invariant to many factors such as shape, size, and scale [40], which are important in the context of heterogeneous traffic.

We use a novel heterogeneous traffic motion and interaction model (HTMI) to predict the position and velocity for each agent in $F_{t+1}$. Our HTMI algorithm is used for two reasons: predicting the collision-free trajectory of each agent and capturing inter-agent interactions between different types of agents. This information is taken into account when computing the next state for an agent.

Finally, we use a simple feature extraction method, where we use a deep CNN to extract novel agent tracking features, called "Deep TA-Features," from both the predicted agents (positions and velocity) obtained from the HTMI model and detected segmented representations obtained from Mask R-CNN. Mask R-CNN creates a pyramid structure of the features called the Feature Pyramid Network (FPN), where it stacks features in layers. Each layer captures a different visual aspect of a traffic agent. For example, the bottom layer may capture the arms and legs of a pedestrian, and the topmost layer may correspond to facial features. With hundreds and thousands of layers in the FPN and each layer encoding some visual aspect like doors or tires, we generate the Deep TA-Features. These features are able to capture the heterogeneity in the traffic. For instance, a feature vector of a truck is easily distinguishable from a pedestrian.

This facilitates easier matching of features of the truck in two successive frames, thereby, improving tracking accuracy. Finally, we match these features using the Cosine metric [41] and IOU overlap (percentage of bounding area overlap) measured based on the Hungarian algorithm [42]. The ID of the detected agent in $F_t$ is matched with a predicted measurement in $F_{t+1}$ and assigned to this agent.

### A. Agent Detection Using Mask R-CNN

A key issue is to detect each different agent in the frame. The projections of the agents are shaped differently. Mask R-CNN uses a Feature Pyramid Network [40] that is ideal for heterogeneous agent tracking as it generates Deep TA-Features (we explain these features in detail in section III-C) that are invariant to multiple factors such as shape and size of heterogeneous agents. In addition, Mask R-CNN successfully detects agents in challenging environments such as nighttime traffic with different lighting conditions, traffic with agents far away in the image. We also observe that R-CNN can also handle jittery camera motion (e.g., cameras mounted to an autonomous vehicle), and low-resolution images.

Our algorithm starts by detecting all the agents in the first frame. Mask R-CNN produces a set of bounding boxes, $B = \{B_{d_j} \mid \exists d_j \in \mathcal{H}\}$, where $\langle x, y, w, h, s, r \rangle$ denote the top left corner, width, height, scale, and aspect ratio of $B_{d_j}$, respectively.

Each bounding box has a corresponding mask for the agent it contains. A mask is a boolean array, each element of which is either true or false, depending on whether the pixel in that location belongs to the agent or not. We create a white background and super impose the pixel-wise segmented agent onto the background using the mask.

Let $\mathcal{W} = \{w_{d_j} \mid d_j \in \mathcal{H}\}$ be the set of white canvases where each canvas, $w_{d_j} = [1]_{w \times h}$, $w$ and $h$ are the width and height of each $B_{d_j}$ and $M_{d_j}$ be the mask for $d_j$. Then, $\mathcal{U} = \{w_{d_j} \langle M_{d_j} \rangle \mid w \in \mathcal{W}, M \in \mathcal{M}, d_j \in \mathcal{H}\}$, is the set of segmented representations for each agent. In Proposition III.2, we show that appearance features generated from segmented representations outperform appearance features generated from bounding boxes that are produced by Faster R-CNN [43].

### B. Heterogeneous Traffic Motion and Interaction (HTMI)

In order to track dense traffic, we need a model that takes into account the interaction between different types of agents with varying shape and dynamics. Moreover, high traffic density

---

**TABLE I: Notation and symbols used in the paper.**

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a_i$</td>
<td>$i^{th}$ agent</td>
</tr>
<tr>
<td>$d_j$</td>
<td>$j^{th}$ detected agent (ie, it has a bounding box)</td>
</tr>
<tr>
<td>$F_t$</td>
<td>current frame at timestep $t$</td>
</tr>
<tr>
<td>$\mathcal{A}$</td>
<td>set of all agents in the current frame, $F_t$</td>
</tr>
<tr>
<td>$\mathcal{H}$</td>
<td>set of all detected agents in the current frame, $F_t$</td>
</tr>
<tr>
<td>$\mathcal{M}_t$</td>
<td>subset of all detected agents in the current frame, $F_t$ that are within a circular region around agent $a_i$</td>
</tr>
<tr>
<td>$B_{d_j}$</td>
<td>bounding box for detected agent $d_j$</td>
</tr>
<tr>
<td>$p_i \equiv (u_i, v_i)$</td>
<td>position of $a_i$, similarly defined for $a_k$</td>
</tr>
<tr>
<td>$v_i \equiv (u_i, v_i)$</td>
<td>velocity of $a_i$, similarly defined for $a_k$</td>
</tr>
<tr>
<td>$f_{a_i}$</td>
<td>feature vectors of the predicted agent, $a_i$</td>
</tr>
<tr>
<td>$d_j$</td>
<td>feature vectors of the segmented representation, $d_j$</td>
</tr>
<tr>
<td>$\cos$</td>
<td>cosine metric defined by $1 - p^T q$</td>
</tr>
<tr>
<td>$t_i$</td>
<td>state of an agent $a_i$ at time $t$, includes position, velocity, and preferred velocity</td>
</tr>
</tbody>
</table>

**Fig. 3:** We show examples of common interactions that take place in dense traffic between heterogeneous agents. In this image, we observe two person-rickshaw interactions and one person-person interaction.

---

**Algorithm:**

1. **Detect Agents:** Use Mask R-CNN to detect all agents in the first frame. Each detected agent has a bounding box and a corresponding mask.
2. **Feature Extraction:** Use a deep CNN to extract features for each agent. These features are invariant to shape, size, and scale.
3. **HTMI Model:** Use a heterogeneous traffic motion and interaction model (HTMI) to predict the position and velocity for each agent in the next frame. The HTMI model uses detected agents and their features to predict future positions.
4. **Feature Matching:** Match predicted features with detected features using the Cosine metric and IOU overlap.
5. **Update Tracking:** Update the tracking of each agent based on the matched features.

---

**Implementation Details:**

- **Training Data:** Use real-world traffic data with heterogeneous agents.
- **Model Architecture:** Use Mask R-CNN with a Feature Pyramid Network (FPN).
- **Optimization:** Use an optimizer like Adam with a learning rate of 0.001.
- **Evaluation:** Evaluate the model using accuracy metrics like IoU overlap and Cosine metric.

---

**Conclusion:** The proposed approach is effective in tracking dense traffic with heterogeneous agents. The HTMI model captures the motion and interactions of agents, improving tracking accuracy. Future work includes improving the model's ability to handle more complex interactions and environments.

---

**Acknowledgments:** This research was supported by [Funding Agency] grants. We thank [Collaborators] for their contributions.

---

**References:**


---

**Appendix:**

- **Equations and Definitions:**
  - **Cosine Metric:** $\cos = 1 - p^T q$
  - **IOU Overlap:** $\text{IOU} = \frac{\text{Area}(A \cap B)}{\text{Area}(A \cup B)}$

---

**Figures:**

- **Fig. 2:** An overview of our approach.
- **Fig. 3:** Examples of common interactions in dense traffic.

---

**Tables:**

- **Table I:** Notation and symbols used in the paper.
increases the probability of collisions with other agents. In order to accurately track heterogeneous agents in dense videos, we need to take into account collision avoidance behavior as well as inter-agent interactions. We present a novel HTMI model that takes into account:

- Reciprocal collision avoidance [25] with car-like kinematic constraints for trajectory prediction and collision avoidance.
- Heterogeneous agent interaction that predicts when two or more agents will interact with each other in the near future.

1) Collision Avoidance: We represent each agent as, $\Psi_{t,i} = [u, v, u_\text{pref}, v_\text{pref}]$, where $u, v, u_\text{pref}, v_\text{pref}$ represent the top left corner of the bounding box, their velocities, and the preferred velocity respectively. The collision avoidance problem can be formally stated as $\Psi_{t+1,i} = f(\Psi_{t,i})$, where $f$ is the RVO function that is used for velocity computation.

The computation of the new state, $\Psi_{t+1,i}$, is expressed as an optimization problem for each agent according to the RVO collision avoidance constraints [25]. If the RVO or ORCA constraints forbid an agent’s preferred velocity, that agent chooses the closest velocity that lies in the feasible region:

$\nu_{\text{RVO}} = \arg\max_{\nu \in \text{ORCA}} ||\nu - \nu_{\text{pref}}||$  \hspace{1cm} (1)

We can handle kinematic and dynamic constraints based on the control obstacles [44], which is a generalization to RVO formulation. This velocity, $\nu_{\text{RVO}}$, is then used to calculate the new position of a traffic agent.

The difference in shapes, sizes, and aspect ratios of agents motivate the need to use appearance-based feature vectors as heterogeneity promotes differentiation between different agents on the basis of appearance. In order to combine object detection (R-CNN) with RVO, we modify the state vector, $\psi_{t,i}$, to include bounding box information by setting the position to the centers of the bounding boxes. Thus

$u \leftarrow \frac{u + w}{2}, v \leftarrow \frac{v + h}{2}$.

2) Heterogeneous Traffic Interactions: When two traffic agents interact, they move towards each other and come in close proximity. We assume that two agents intend to interact if they have been moving towards each other for some time $\tau$. We now present a condition for two agents $a_i, a_k$ to be able to interact and use this information as part of our prediction module (see Figure 5). Given a traffic agent $a_i$, the slope of $v_{\text{pref},i}$ is $\tan \theta_i$. In dense traffic, each agent has a limited space in which they can steer, or turn. We denote this steering angle as $\phi_i$. We define a circular region, $\zeta$ of radius $\rho$, centered around $a_k$ that represents the personal space of $a_k$. Based on the direction of current preferred velocity and the steering angle, the space in which $a_i$ can freely move is defined by a 2D cone, which we denote as $\gamma$. We denote the extreme rays of $\gamma$ as $r_1$ and $r_2$. $\perp \gamma_2$ denotes the smallest perpendicular distance between two geometric structures, $g_1$ and $g_2$.

Our heterogeneous interaction module assumes that if the projected cone of $a_i$, defined by extending $r_1$ and $r_2$, intersects with the $\zeta$, then $a_i$ can interact with $a_k$. Based on this assumption, it is sufficient to check for either one of two conditions: intersection of $\zeta$ with either $r_1$ or $r_2$ (if either ray intersects, then the entire cone intersects $\zeta$) and $\zeta \subset \gamma$ (if $\zeta$ lies in the interior of $\gamma$, see Figure 5). $r_1, r_2$ are parameterized based on their slopes $\tan \delta_i$, where $\delta = \theta_i + \phi_i$ if $\perp r_1 \geq \perp r_2$, else $\delta = \theta_i - \phi_i$. The resulting equation of $r_1$ (or $r_2$) is $(Y - v_k) = \tan \delta (X - u_k)$ and the equation of $\zeta$ is $(X - u_k)^2 + (Y - v_k)^2 = \rho^2$. Solving both equations, we obtain

$$X^2 \left( \sec^2 \delta \right) + 2X \left( \tan \delta (v_i - v_k) - u_k - u_i \tan^2 \delta \right) + \left( u_k^2 + (v_i - v_k)^2 + u_i^2 \tan^2 \delta - 2 \tan \delta (v_i - v_k)u_i - \rho^2 \right),$$  \hspace{1cm} (2)
Intersection occurs if the discriminant, $\Omega_1$, of (2) $\geq 0$. This provides us with one condition for the occurrence of an interaction between $a_i$ and $a_k$.

Moreover, we observe that if $\zeta$ lies in the interior of $\gamma$, then $p_k$ lies on the opposite sides of $r_1$ and $r_2$ which is modeled by the following equation:

$$ r_1(p_k) \cdot r_2(p_k) \leq 0 \quad (3) $$

Solving (3) further provides us the second condition for the occurrence of an interaction between $a_i$ and $a_k$.

$$ \Omega_2 \equiv (v_k - v_i)^2 + (u_k - u_i)^2 \tan \alpha \tan \beta - (v_k - v_i)(u_k - u_i)(\tan 2\theta_i)(1 - \tan \alpha \tan \beta) \leq 0 $$

where $\Omega_1, \Omega_2 : \mathbb{R}^2 \times \mathbb{R}^2 \times \mathbb{R} \times \mathbb{R} \mapsto \mathbb{R}$.

If either $\Omega_1$ or $\Omega_2$ is true, then agents $a_i, a_k$ will move towards each other to interact in the near future. When this happens, we assume that $a_i$ and $a_k$ align their current velocities towards each other. In that case, the time taken for the two agents to be very close (or overlap) with each other is given by:

$$ t = \frac{|p_i - p_k|^2}{|v_i - v_k|^2} $$

If two agents are overlapping (based on the values of $\Omega_1$ and $\Omega_2$), we model them as a new agent with radius $2\varepsilon$. In this case, two agents are classified to be interacting if, $|p_i - p_k|^2 = \varepsilon$. Our approach can be extended to model interactions.

We form a set $\mathcal{I} \subseteq \mathcal{A}$, where $\mathcal{I}$ is the set of all agents $a_\omega$, that are intending to interact with $a_k$. In order to determine the first agent that may interact with $a_k$, we can compute the time taken by that agent that wishes to interact with $a_k$ by: $t_{\text{min}} = \arg \min_{\omega} |\nu_\omega t - p_k| = \rho, a_\omega \in \mathcal{I}$. Agents that are not interacting avoid each other and continue moving towards their destination.

We use our model to formulate when two agents can interact in a densely crowded traffic scene. We model the time taken by that agent to reach the other agent, and we also take into account multiple interactions in the same scene. Using the new positions and velocities after capturing these interactions, HTMI uses RVO to predict new positions and velocities of the interacting agents.

### C. Feature Extraction Using Segmented Representations

We extract simple, yet powerful, features, called “Deep TA-Features”, from segmented shapes, that are obtained from Mask R-CNN, using a convolutional neural network. The architectural and training details of this network are described in [45]. Deep TA-Features are grid-like maps that are stacked in a pyramid-shaped structure. Each layer of this pyramid encodes a visual aspect (for example, facial features or vehicle parts) of a traffic agent. We present a formulation in which Deep TA-Features significantly improve the performance of our algorithm by reducing false negatives, (ground truth agents that are not tracked) and identity switches (incorrect assignment of tracking ID). These often arise in dense scenarios due to increased instances of collisions and interactions.

We define $\mathcal{T}_{i,i} = \{\Psi_{1,i,i}\}$ to be the set of states for a correctly tracked traffic agent, $a_i$ until time $t$. We denote the time since the last update to an agent’s ID as $t_a$. We denote the ID of $a_i$ as $\alpha_i$ and we represent the correct assignment of an ID to $a_i$ as $\Gamma(\alpha_i)$. The threshold for the cosine distance is $\lambda_{i,i.d} = \mathbb{U}[0, 1]$. The threshold for the track age is $t_a$. We denote the probability of an event that uses Mask R-CNN by $\mathbb{P}^M(\cdot)$ and the probability of an event that uses Faster R-CNN by $\mathbb{P}^F(\cdot)$. $\mathbb{P}$ without a superscript denotes general probability. Finally, $\mathcal{T}_{i,i} \leftarrow \{\emptyset\}$ represents the loss of $\mathcal{T}_{i,i}$ by occlusion, collision, or interaction between heterogeneous agents in dense traffic.

The following lemma highlights some properties of these feature vectors. It shows that the Cosine cost of Deep TA-Features is less than the Cosine cost of features generated from prior algorithms like Faster R-CNN. Thus,

**Lemma III.1.** For every pair of feature vectors, $(f^M_{d}, f^F_{d})$ generated from a segmented box and a bounding box, respectively, $l(f^M_{d,j}, f^F_{d,q}) < l(f^F_{d,j}, f^F_{d,q})$.

The proof is described in the report [46].

**Proposition III.2.** Feature vectors extracted from segmented representations decrease the probability of the loss of agent tracks and increase the probability of correct ID assignment to an agent, without additional training or complex optimizations, thereby reducing the number of false negatives and ID switches.

**Proof:** The correct assignment of an ID depends on successful feature matching between the predicted measurement feature and the optimal segmented shape feature. Equivalently,

$$ d(f_{a_i}, h_{\alpha_i}) > \lambda \Leftrightarrow \alpha_i = \emptyset, \forall a_i. $$

Using III.1 and the fact that $\lambda_{i,i.d} = \mathbb{U}[0, 1]$, we get

$$ \mathbb{P}(d(f_{a_i}, h_{\alpha_i}) > \lambda) < \mathbb{P}(d(f_{a_i}, h_{\emptyset}) > \lambda) $$

Using (4), it directly follows that $\mathbb{P}^M(\alpha_i = \emptyset) < \mathbb{P}^F(\alpha_i = \emptyset)$. In our approach, we set $t_a > t_i \land \alpha_i = \emptyset \Leftrightarrow \mathcal{T}_{i,i} \leftarrow \{\emptyset\}$. It follows that,

$$ \mathbb{P}^M(\mathcal{T}_{i,i} \leftarrow \{\emptyset\}) < \mathbb{P}^F(\mathcal{T}_{i,i} \leftarrow \{\emptyset\}) \quad (5) $$

We define the total number of false negatives (FN) as $FN = \sum_{t=1}^{T} \sum_{a_i \in \mathcal{G}} \delta_{T_{i,i}}$, where $\delta_a \in \mathcal{G}$ denotes a ground truth agent in the set of all ground truth agents in $\mathcal{G}$ and $\delta_z = 1$ for $z = 0$ and 0 elsewhere, is a variation of the Kronecker delta function. Using (5), we can say that fewer lost tracks $(\mathcal{T}_{i,i} \leftarrow \{\emptyset\})$ indicate a smaller number of false negatives. Finally, by using lemma III.1 with the association formulation (6) described in the next subsection, it is easy to see that $\mathbb{P}^M(\Gamma(\alpha_i)) > \mathbb{P}^F(\Gamma(\alpha_i))$, which completes the proof.

### D. Feature Matching Using Association Algorithms

Deep TA-Features corresponding to the same agent ID are likely to be more similar, and we use appropriate algorithms.
We measure this similarity in two ways: the Cosine metric \([41]\) and the IOU overlap \([47]\). The task of assigning a new ID to a predicted measurement for \(a_i\) becomes that of computing the optimum detection, \(d^*_{j,i}\); that is, the detection whose feature vector is most similar to \(f_a_i\). This is posed as an optimization problem:

\[
d^*_j,a_i = \arg\min_{d_j} l(f_a_i, f_d_j) |a_i \in A, d_j \in \mathcal{H}_i.
\] (6)

The IOU overlap metric is used together with the cosine metric. This metric builds a cost matrix, \(C\) to measure the amount of overlap of each predicted bounding box with all nearby detection bounding box candidates. \(C(i,j)\) stores the IOU overlap of the bounding box of \(\Psi_{t+1,a_i}\) with that of \(d_j\) and is calculated by

\[
C(i,j) = \frac{B_{\Psi_{t+1,a_i}} \cap B_{d_j}}{B_{\Psi_{t+1,a_i}} \cup B_{d_j}}, d_j \in \mathcal{H}_i.
\]

Matching a detection to a predicted measurement with maximum overlap thus becomes a max weight matching problem that can be solved using the Hungarian algorithm \([42]\).

IV. IMPLEMENTATION AND RESULTS

In this section, describe our implementation and highlight the performance on different datasets.

A. New Dataset and Baseline Results

We use a new dataset that consists of a set of 19 video sequences that contain dense traffic with highly heterogeneous agents with varying viewpoints, camera motions, and at different times of the day. These videos correspond to the highway and urban traffic in the USA, China, and India. Most importantly, ground truth annotations are provided with the dataset.

The key aspects of this dataset are the density and the degree of heterogeneity. Compared to standard traffic datasets such as KITTI \([48]\) and PETS\([49]\), our dataset is denser with a higher degree of heterogeneity (table IV). This includes challenging sequences such as nighttime traffic with heavy glare from oncoming traffic, scenes with jittery camera motion, and scenes with agents far away from the camera.

We provide baseline results on our dataset using our tracking algorithm and demonstrate a high average accuracy of 72.02\% (table III) operating at an average speed of 0.4s on a single GPU. However, our algorithm can be optimized further by sub-sampling and using parallelization to increase the frame rate. We classify accuracy according to the CLEAR metrics \([50]\) as \(1 - (FN + FP + IDS\)/GT\), where FN, FP, IDS, and GT correspond to the false negatives, false positives, ID switches, and ground truth, respectively. We do not count stationary agents such as parked vehicles in our formulation. Objects such as traffic signals are considered false positives. We take into account all possible road agents. Thus, negatives that include agents such as carts and animals are also considered as false negatives.

B. Comparison: Online Methods on Standard Benchmarks

We also evaluate our tracking algorithm on the KITTI-16 sequence and compare with online methods on the MOT benchmark \([51]\). We only compare our algorithm with methods that have an average rank higher than ours on the MOT benchmark. We achieve the lowest number of false negatives and identity switches, which is a direct consequence of III.2.

V. CONCLUSION AND FUTURE WORK

We present a realtime, novel, end-to-end tracking algorithm called DeepTAgents for traffic agents, including vehicles, bicycles, pedestrians, etc. in highly dense scenarios. We use tracking-by-detection paradigm and present a new motion and interaction model (HTMI). Our approach is general and evaluated on a new dense video datasets. Our approach is 4X faster than prior tracking algorithms and we observe considerable improvement in accuracy.

There are many avenues for future work. Besides accelerating the performance, we can further improve the accuracy. This includes improved HTMI model, that uses a more accurate model for dynamics as well as improving the accuracy of TA-feature detection. We would like to further evaluate the performance on complex traffic videos.

REFERENCES

TABLE II: Evaluation on the KITTI-16 benchmark with online methods that use public detections that have an average rank higher than ours. Bold is best, blue is second best. References for these papers can be found at https://motchallenge.net/. Arrows (↑, ↓) indicate the direction of better performance. Our algorithm is >4X faster and 7% more accurate in the number of correctly tracked agents.

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Length</th>
<th>Day/Night</th>
<th>Camera Motion</th>
<th>Accuracy (MOTA)</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRAF12</td>
<td>0:35</td>
<td>Day</td>
<td>Moving</td>
<td>73.2%</td>
</tr>
<tr>
<td>TRAF15</td>
<td>0:29</td>
<td>Day</td>
<td>Moving</td>
<td>68.34%</td>
</tr>
<tr>
<td>TRAF17</td>
<td>0:19</td>
<td>Day</td>
<td>Moving</td>
<td>65.94%</td>
</tr>
<tr>
<td>TRAF18</td>
<td>0:33</td>
<td>Day</td>
<td>Moving</td>
<td>77.50%</td>
</tr>
<tr>
<td>TRAF19</td>
<td>0:17</td>
<td>Day</td>
<td>Moving</td>
<td>67.15%</td>
</tr>
<tr>
<td>TRAF20</td>
<td>1:00</td>
<td>Day</td>
<td>Moving</td>
<td>81.32%</td>
</tr>
<tr>
<td>TRAF21</td>
<td>0:31</td>
<td>Day</td>
<td>Moving</td>
<td>79.92%</td>
</tr>
<tr>
<td>TRAF22</td>
<td>0:45</td>
<td>Day</td>
<td>Moving</td>
<td>66.87%</td>
</tr>
<tr>
<td>TRAF23</td>
<td>0:19</td>
<td>Night</td>
<td>Moving</td>
<td>67.92%</td>
</tr>
</tbody>
</table>

TABLE III: We highlight the tracking accuracy of DeepTAgent on 9 of our 19 high resolution (720p) video sequences of dense heterogeneous traffic with different types of agents. All sequences have been carefully annotated with ground truth labels following a strict protocol. The sequences are categorized according to viewpoint, time of the day, camera motion, and length. Tracking accuracy on all the 19 sequences are available on our website.

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Length</th>
<th>Day/Night</th>
<th>Camera Motion</th>
<th>Accuracy (MOTA)</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRAF11</td>
<td>0:45</td>
<td>Day</td>
<td>Moving</td>
<td>79.92%</td>
</tr>
<tr>
<td>TRAF12</td>
<td>0:35</td>
<td>Day</td>
<td>Moving</td>
<td>73.2%</td>
</tr>
<tr>
<td>TRAF15</td>
<td>0:29</td>
<td>Day</td>
<td>Moving</td>
<td>68.34%</td>
</tr>
<tr>
<td>TRAF17</td>
<td>0:19</td>
<td>Day</td>
<td>Moving</td>
<td>65.94%</td>
</tr>
<tr>
<td>TRAF18</td>
<td>0:33</td>
<td>Day</td>
<td>Moving</td>
<td>77.50%</td>
</tr>
<tr>
<td>TRAF19</td>
<td>0:17</td>
<td>Day</td>
<td>Moving</td>
<td>67.15%</td>
</tr>
<tr>
<td>TRAF20</td>
<td>1:00</td>
<td>Day</td>
<td>Moving</td>
<td>81.32%</td>
</tr>
<tr>
<td>TRAF21</td>
<td>0:31</td>
<td>Day</td>
<td>Moving</td>
<td>79.92%</td>
</tr>
<tr>
<td>TRAF22</td>
<td>0:45</td>
<td>Day</td>
<td>Moving</td>
<td>66.87%</td>
</tr>
<tr>
<td>TRAF23</td>
<td>0:19</td>
<td>Night</td>
<td>Moving</td>
<td>67.92%</td>
</tr>
</tbody>
</table>

TABLE IV: We pick the top two sequences of various traffic datasets and compare heterogeneity with ours (TRAF11 and TRAF12). We observe a high variety of traffic agents.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Person</th>
<th>Animal</th>
<th>Vehicle</th>
<th>AP</th>
<th>RAR</th>
<th>F1</th>
<th>MOT(%)</th>
<th>ML(%)↓</th>
<th>MOJA(%)↑</th>
<th>IDSW(%)↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>KITTI-16</td>
<td>5.6</td>
<td>0.3</td>
<td>0.1</td>
<td>831</td>
<td>17.6</td>
<td>118</td>
<td>40.7</td>
<td>41.2</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>PETSOV (2)</td>
<td>28.5</td>
<td>41.5</td>
<td>32.0</td>
<td>12.2</td>
<td>5.9</td>
<td>17.6</td>
<td>46.3</td>
<td>10</td>
<td>10</td>
<td></td>
</tr>
</tbody>
</table>


