APPENDIX

I. Car Following Filter

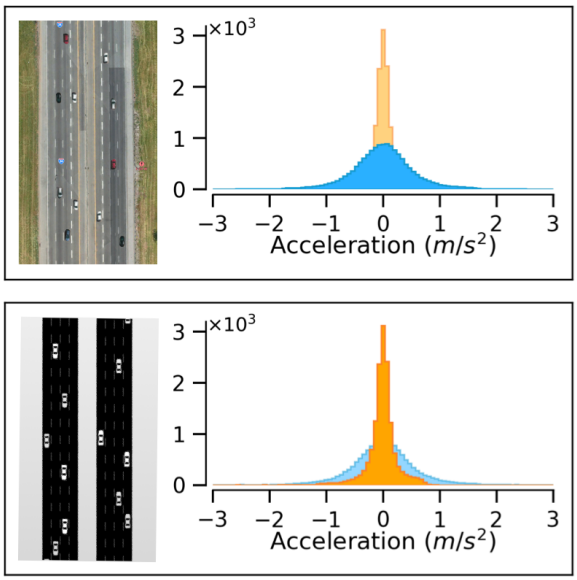

Fig. 3: Instantaneous accelerations observed during car-following at densities ranging from 70 to 150 veh/km. TOP: Real-world data from the I-24 MOTION v1.0.0 dataset reveals a distribution with long tail extending upto $[-3, 3]$ m/s^2. BOTTOM: IDM model in simulation produces accelerations mostly constrained within $[-0.5, 0.5]$ m/s^2.

We analyze the I-24 MOTION v1.0.0 dataset [17] with study length $= 6.75$ km and study time $= 4$ h. The dataset contains various vehicle types such as semi-trailers, mid-sized trucks, motorbikes, and cars in different traffic situations such as approaching standing traffic, lane change, and free flow. We extract car-following trajectories by filtering the data with the following exceptions:

1) An ego car is following another car (has a leader).
2) Leader and ego car are in the same lane for 5 s or more.
3) Ego car's speed is greater than 10% of the speed limit (avoid approach to stationary traffic).
4) Ego car's space headway is less than 124 m, applying 4 s rule at speed limit to avoid free flow conditions.

II. Model based Robot Vehicles

Bilateral Control Module (BCM): BCM [6] uses information about both following and leading vehicles to obtain a linear model whose acceleration is given by:

$$a = k_d \cdot \Delta_d + k_v \cdot (\Delta v_l - \Delta v_f) + k_c \cdot (v_{des} - v) \quad (1)$$

where Δ_d, Δv_l, Δv_f, v_{des}, and v, represent the difference of distance to the leader and distance to the follower, the difference in velocity of to the leader, the difference in velocity to the follower, the set desired velocity, and the current velocity of the of vehicle respectively. $k_d = 1$, $k_v = 1$, and $k_c = 1$ are gain parameters.

Linear Adaptive Cruise Control (LACC): LACC is an improvement on cruise control systems that allows vehicles to automatically maintain a safe distance or speed without the need for communication. The constant time-headway model described by Rajamani [7] employs a first-order differential equation approximation. The control acceleration at time t is given by:

$$a_t = (1 - \frac{\Delta t}{\tau}) \cdot a_{(t-1)} + \frac{\Delta t}{\tau} a_{cmd,(t-1)} \quad (2)$$

$$a_{cmd} = k_1 \cdot e_x + k_2 \cdot \Delta v_l \quad (3)$$

$$e_x = s - h \cdot v \quad (4)$$

where $k_1 = 0.3, k_2 = 0.4$ are design paramters, e_x is the gap error, s is the space headway, Δv_l is the velocity difference to the leader, $h = 1$ is the desired time gap, Δt is the control time-step, and $\tau = 0.1$ is the time lag of the system.

III. Heuristic based Robot Vehicles

FollowerStopper (FS): FS [5] is a velocity controller that closely tracks a fixed average velocity without colliding. The RV travels at slightly lower speed and opens up a gap to the vehicle ahead, allowing it dampen oscillations and brake smoothly when needed. The command velocity is given by:

$$v_{cmd} = \begin{cases} 0, & \text{if } \Delta x \leq \Delta x_1 \\ v \frac{\Delta x - \Delta x_1}{\Delta x_2 - \Delta x_1}, & \text{if } \Delta x_1 < \Delta x \leq \Delta x_2 \\ v + (U - v) \frac{\Delta x - \Delta x_2}{\Delta x_3 - \Delta x_2}, & \text{if } \Delta x_2 < \Delta x \leq \Delta x_3 \\ U, & \text{if } \Delta x_3 < \Delta x \end{cases} \quad (5)$$

where $v = \min(\max(v_{\text{lead}}, 0), U)$ is the speed of the leading vehicles, Δx is the headway of the RV, and the boundaries are defined as:

$$\Delta x_k = \Delta x_k^0 + \frac{1}{2d_k}(\Delta v_-)^2, \quad k = 1, 2, 3 \quad (6)$$

The model parameters Δx_0, Δv_-, and d_k determine the spacing between vehicles and the sensitivity of the RV to changes in velocity, while U is the desired velocity.

Proportional-integral with saturation (PIwS): PIwS [5] estimates the average equilibrium velocity of all vehicles in the network using its own historical velocity average (U) and then drives at the estimated velocity. The target velocity is calculated as:

$$v_{target} = U + v_{catch} \times \min\left(\max\left(\frac{\Delta x - g_l}{g_u - g_l}, 0\right), 1\right) \quad (7)$$

which is then used to calculate the command velocity at time $t + 1$ as:

$$v_{cmd,t+1} = \beta_t(\alpha_t v_{target,(t)} + (1 - \alpha_t) v_{lead,(t)}) \quad (8)$$

$$+(1 - \beta_t) v_{cmd,(t)} \quad (9)$$

where v_{catch} is the catch-up velocity, Δx is difference between the position of the RV and its leader, g_l and g_u represent the lower and upper threshold distances, respectively and α_t and β_t represent the weight factors for target velocity v_{target} and command velocity v_{cmd} respectively. Finally, v_{lead} represents the velocity of the lead vehicle.

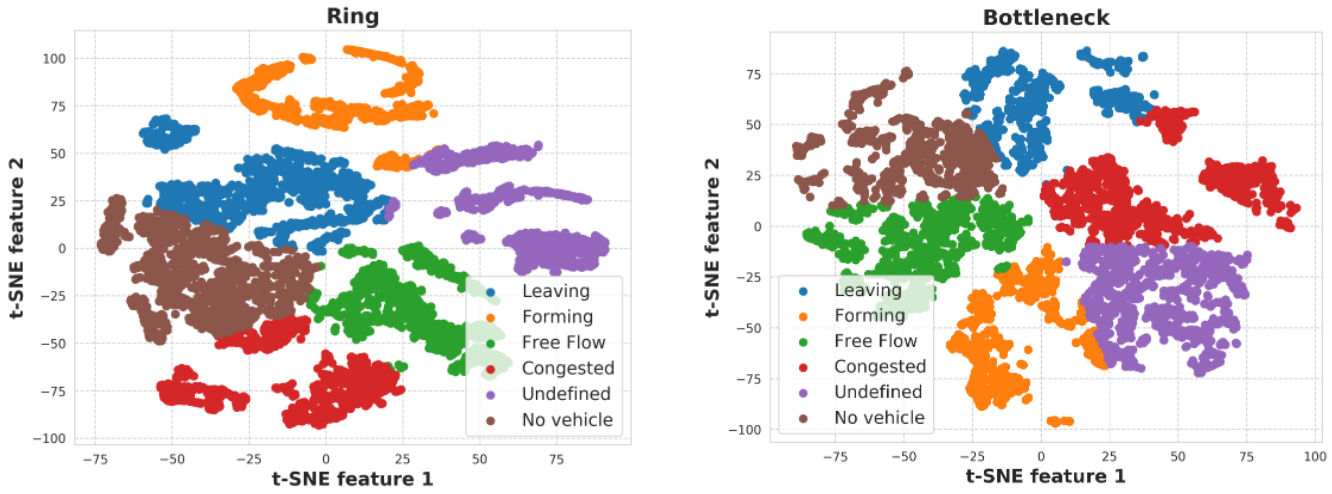

Fig. 4: K-means clustering with t-SNE on the data collected to train the CSC in each environment. LEFT: In the Ring, the clusters are spread out and distinct suggesting that the data is easily classifiable. RIGHT: In the Bottleneck, clusters are overlapping suggesting more complex interactions between vehicles, possibly due to the presence of zipper lanes where vehicles can (abruptly) merge in front of a RV.

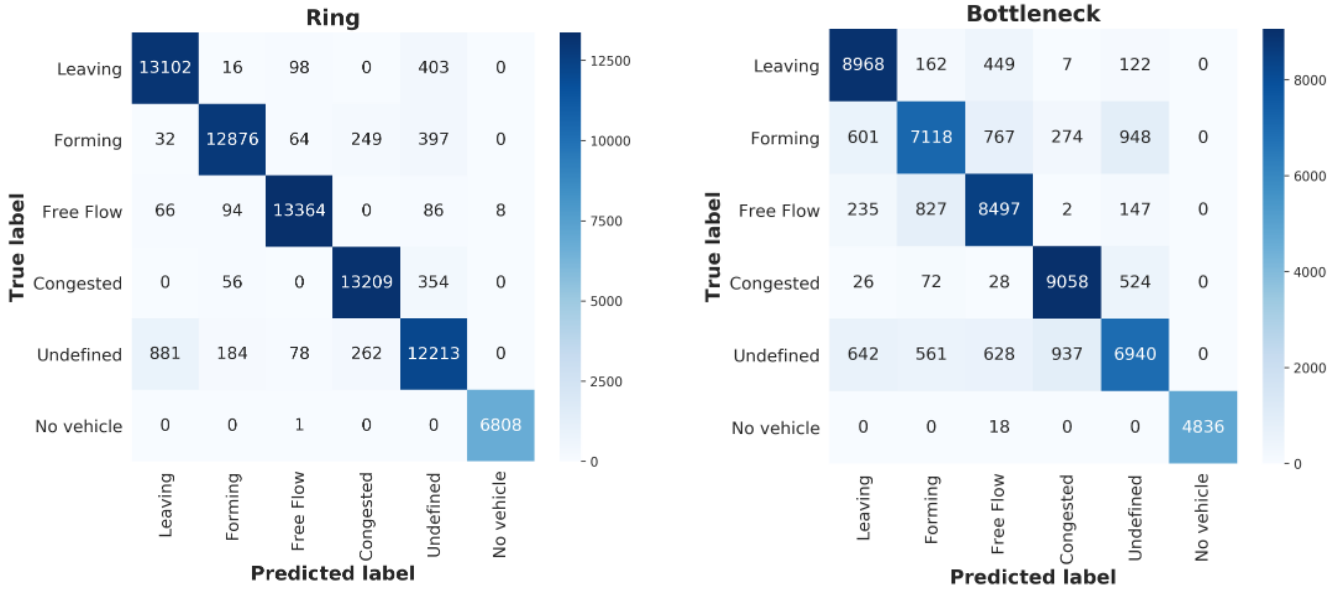

Fig. 5: Confusion Matrix of a trained CSC in the Ring (LEFT) the Bottleneck (RIGHT) on the validation set.

IV. Reinforcement Learning (RL) Benchmarks

For RL with only local observations (RL + L), we follow the experimental setup and hyper-parameters from Wu et al. [8] obtaining a policy that closely matches the stabilization time and average velocity during stabilization, at various densities within 3% of the reported original work. Whereas for RL with global observations (RL + G), we follow the same set of parameters as Vinitsky et al. [11] to obtain a policy with outflow within 1% of the reported original work.

V. Congestion Stage Classifier (CSC)

One CSC each is trained in the Ring and the Bottleneck with independent data collection on both networks. For each RV, (position, velocity) of all the vehicles in its local zone (set to 50 m) is collected. The K-means clustering of collected data for all 6 classes in both environments is shown in Figure 4. As the data collected is sequential in nature and CSC prediction is made a number of time-steps into the future, a time offset of 10 time-steps is chosen balancing usefulness and accuracy (to illustrate, prediction of congestion stage 100 time-steps in the future would be very useful, however, is not very accurate, whereas prediction of congestion state 1 time-step into the future is almost 100% accurate but not very useful). After windowing, the data consists of both transitions from one class label (congestion stage) to another as well as no such transition between time-step t to $t + 10$. To train the CSC, data is sampled in a way to ensure a balanced representation of transition/ non-transition as well as all 6 classes. Further, as solely based on "No vehicle" state at time-step t, we cannot predict what the congestion stage will be in time-step $t + 10$, hence the data collected for "No vehicle" class where transitions to another

Category	Parameter	Value
Ring Simulation	Time Step (Δt)	0.1
	Simulation Horizon (T)	4500
	Warmup Time-steps	2500
	Speed Limit (m/s)	30
	Initial Speed (m/s)	0
Bottleneck Simulation	Time Step (Δt)	0.5
	Simulation Horizon (T)	1300
	Warmup Time-steps	100
	Speed Limit (m/s)	17
	Initial Speed (m/s)	6
	Inflow Rate (veh/hr)	3600
PPO Algorithm	Learning Rate (α)	0.00005
	Discount Factor (γ)	0.999
	GAE Estimation (λ)	0.97
	KL Divergence Target	0.02
	Entropy Coefficient Initial	0.1
	Entropy Coefficient Final	0.01
	Value Function Clip Param	20
	SGD Iterations	2
Ring CSC	Neural Network	$32, 16, 16$
	Batch size	32
	Learning rate	0.01
	Epochs	50
Bottleneck CSC	Neural Network	$32, 16, 16$
	Batch size	256
	Learning rate	0.001
	Epochs	100
Policy Networks	Our leader RV in the Ring	$64, 32, 16$
	Our follower RV in the Ring	$64, 32, 16$
	Our RV in the Bottleneck	$32, 16, 8$
	RL + L	$32, 32, 32$
	RL + G	$256, 256$

TABLE III: Simulation settings, parameters of the Proximal Policy Optimization (PPO) algorithm and Congestion Stage Classifier (CSC) and hidden layer dimensions of various policy networks.

class occur after 10 time-steps is discarded and replaced with synthetic data with various possibilities of RV (position, velocity) with no other vehicles in front.

After training, the accuracy of CSC in the Ring is 95.5% whereas in the bottleneck it is 85.2%. The confusion matrix is shown in Figure 5. The CSC only observes the HVs downstream in the traffic within the same lane, however, the Bottleneck consists of zipper lanes, where traffic merges from adjacent lanes contribute to sudden behaviors that CSC cannot anticipate, contributing to lower accuracy in the Bottleneck. The CSC training parameters are provided in Table. III. Lastly, among the various ways in which a CSC is leveraged in this work, our *efficiency* RV in both the Ring and the Bottleneck constantly estimate free-flow traffic speed (a free-flow stage is determined based on CSC output); RV acceleration is subsequently set to zero (maintaining constant speed) when the RV's speed exceeds this estimated free-flow speed, thereby successfully optimizing fuel economy in the Ring.