Supplementary Materials for COOPERNAUT: *End-to-End Driving with Cooperative Perception for Networked Vehicles*

Jiaxun Cui[*1] Hang Qiu[*2] Dian Chen[1] Peter Stone[1,3] Yuke Zhu[1]
[1]The University of Texas at Austin [2]Stanford University [3]Sony AI
cuijiaxun@utexas.edu, hangqiu@stanford.edu, {dchen, pstone, yukez}@cs.utexas.edu

1. Implementation Details

1.1. Model Hyperparameters

When the ego vehicle receives messages from more than three neighboring vehicles, we randomly select messages from 3 of the 6 nearest neighboring vehicles within a 40-meter range. Every raw LiDAR point cloud may contain up to 65,536 coordinates. To down-sample the point cloud, we merge the points through max-pooling within each cell of a voxel grid of size $0.5m \times 0.5m \times 0.5m$. After ground plane removal, we randomly select 2,048 points as the encoder input. All the neighbors encode their processed point clouds locally by the 3-block Point Transformer encoder and send the message of size $128 \times (128, 3)$ and warp the coordinates to the ego frame. In our 3-neighbor case, the received message size is $384 \times (128, 3)$. We filter out the out-of-range messages and apply voxel max-pooling to these messages for size reduction. We aggregate the merged representations by another block of down sampling and point transformer. After global max pooling, the features are concatenated with the ego speed feature before passing to the fully connected layers that has has 256 and 64 hidden units. The multi-head attention layers in the Point Transformer uses an embedding size of 32, and the kNN considers 16 nearest neighbors.

To comply with the speed limit rules, we apply a PID controller and use the output (denoted as B_p) as a reference for brake during the interaction with the environment. The control decision $(a_{\text{throttle}}, a_{\text{brake}}, a_{\text{steer}})$ follows:

$$a_{\text{throttle}} = \begin{cases} 0 & \text{if } T \leq B \text{ or } B_p > 0 \\ T & \text{otherwise} \end{cases}$$

$$a_{\text{brake}} = \begin{cases} B & \text{if } T \leq B \\ B_p & \text{if } T > B \text{ and } B_p > 0 \\ 0 & \text{otherwise} \end{cases}$$

$$a_{\text{steer}} = S$$

[*] Equal contribution. Correspondence: cuijiaxun@utexas.edu, hangqiu@stanford.edu, yukez@cs.texas.edu

Parameter	Value
raw LiDAR point size	65,536
lidar range(radius)	$70m \times 70m \times 5m$
lidar scanning frequency	10 Hz
voxel size	$0.5m \times 0.5m \times 0.5m$
npoints	2,048
kNN neighbors	16
transformer dimensions	32
control module MLP structure	[128+8, 256, 64, 3]
point transformer blocks	3
max number of neighbors	3

Table 1. Model Parameters

Detailed model parameters are listed in the Table 1.

1.2. Training Details

Our model training consists of two stages: behavior cloning and DAgger. We first train every scenario-specific model for 100 epochs by behavior cloning on 12 trajectories, with 3 of them collected under accident-prone configurations (*i.e.*, with an occluded collider vehicle inserted) and 9 of them being normal driving trajectories. The weights of neural networks are optimized with the Adam optimizer with an initial learning rate 10^{-4}, weight decay 10^{-5}, and batch size 32. Then the final policy of behavior cloning serves as the initial student policy at the second stage for DAgger. We collect 4 new trajectories every 5 epochs using a sampling policy with $\beta_0 = 0.8$. The newly-collected data, which has a normal-to-accident-prone ratio of 3:1, are aggregated to the DAgger training dataset. We run DAgger training for 105 epochs (*i.e.*, 21 iterations). We evaluate the final policy after the last iteration. The training time varies among different models. The average wall time needed for training COOPERNAUT is roughly 60 hours with a 24GB Nvidia GeForce 3090 GPU and 16 Intel(R) Core(TM) i9-10900KF @ 3.70GHz CPUs. Table 2 lists the important training parameters.

Parameter	Value
BC batch size	32
BC training epochs	100
BC sample size	12 trajectories
DAgger batch size	32
DAgger training epochs	105
DAgger β_0	0.8
DAgger sampling frequency	4 trajectories/5 epochs
DAgger sample size	88 trajectories

Table 2. Training Parameters

1.3. Trajectory Generation

For every trajectory generated, we stop collecting more data for this trajectory if the agent crashes into any entity in the environment or stagnates for more than 30 seconds (simulator time) or exceeds the total time limit per scenario.

2. Dataset

AUTOCASTSIM is highly scalable to the computing resources. In our experiments, simulation processes for data collection are parallelized by the Ray library [1]. Users can specify the desired traffic densities, ranges of dangerous environment configurations (*e.g.*, other vehicles' speed and collision location), and the number of desired trajectories. Then AUTOCASTSIM procedurally generate environments with the specifications and produce the desired number of trajectories. The datasets for the training pipeline are split into the following partitions:

Train for behavior cloning (BC) (12 expert trajectories, including 3 accident-prone and 9 normal),

Validation for validation during training BC and DAgger (3 accident-prone trajectories),

Expert (81 accident-prone trajectories generated by the expert) collected using a fixed set of random seeds to control the environment randomness for fair comparisons among the expert and different studied methods,

DAgger for online training with DAgger (88 trajectories generated by the DAgger sampling policy, including 22 accident-prone and 66 normal).

3. Latency

COOPERNAUT achieves a 90ms latency of the entire inference pipeline. The Point Encoder takes up 80ms for every vehicle on Nvidia GeForce 3090 GPU and Intel(R) Core(TM) i9-10900KF CPU @ 3.70GHz CPUs.

4. Compression

The intermediate representation is of shape $128 \times (128 + 3)$, represented by the 32-bit float numbers. The package size is about 0.51 Mb and requires a 5.1Mbps bandwidth if we transmit packages at a 10Hz frequency, which satisfies the bandwidth constraints of V2V channels. Nonetheless, it is possible to apply lossless compression algorithms to further reduce the message sizes.

5. Trajectories

Figure 1a, 1b, and 1c illustrate some examples of critical frames in the trajectories of different scenarios between the No V2V Sharing baseline and COOPERNAUT.

- **Left Turn.** The ego vehicle tries to turn left on a left-turn yield light but encounters another truck in the opposite left-turn lane, blocking its view of the opposite lanes and potential straight-going vehicles. In the Left Turn yielding scenario (Figure 1a), COOPERNAUT is shown to avoid the collision by proactively stopping at the junction and does not turn left until there is no close opposite going cars.
- **Overtaking.** A truck is blocking the way of a sedan in a two-way single-lane road with a dashed yellow lane divider. The truck is also impeding the sedan's view of the opposite lane. An autonomous agent has to make a lane change maneuver to overtake. In Figure 1b, COOPERNAUT is shown to wait for a safe gap between the opposite traffic flow before making the lane-change maneuver, while the No V2V Sharing policy aggressively changes lane.
- **Red Light Violation.** The ego vehicle is crossing the intersection when there is another vehicle rushing the red light. LiDAR fails to detect the other vehicle because of the lined-up cars waiting for the left turn. In Figure 1c, COOPERNAUT is shown to detect the dangerous traffic violator and brake before possible collisions, while the No V2V Sharing policy proceeds straight to collide with the traffic violator.

6. Datasets and Codebases

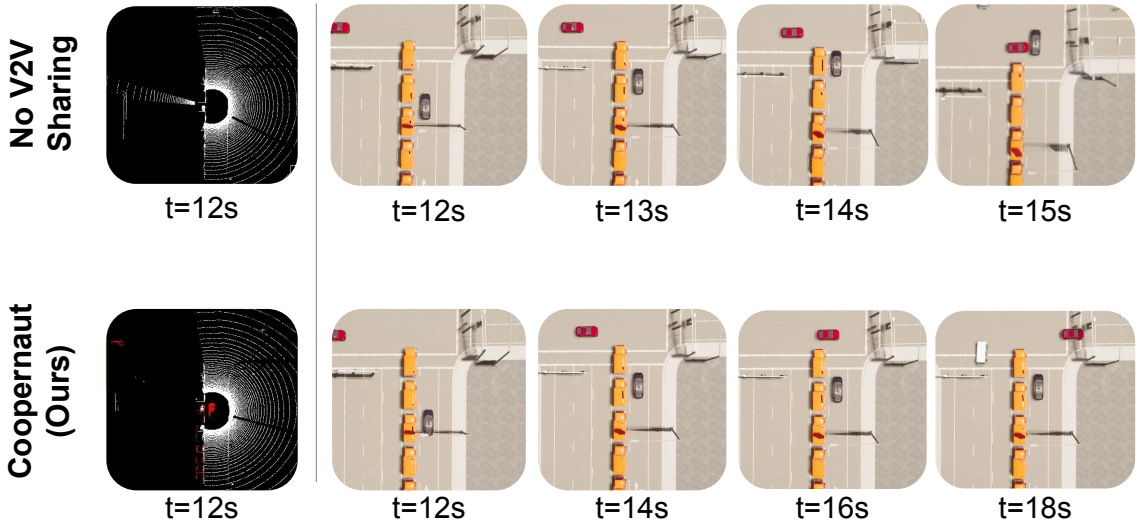
We provide additional information and resources on our project website: https://ut-austin-rpl.github.io/Coopernaut/. Links to the training datasets and codebases of COOPERNAUT and AUTOCASTSIM can be found on this website.

References

[1] Philipp Moritz, Robert Nishihara, Stephanie Wang, Alexey Tumanov, Richard Liaw, Eric Liang, Melih Elibol, Zongheng Yang, William Paul, Michael I Jordan, et al. Ray: A distributed framework for emerging ai applications. In *13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18)*, pages 561–577, 2018. 2

(a) Left Turn scenario.

(b) Overtaking.



(c) Red Light Violation.

Figure 1. Example trajectories of No V2V Sharing model and COOPERNAUT under 3 scenarios.