Appendix

A Theoretical Background of Importance Sampling

Importance sampling is a variance-reduction technique for Monte Carlo sampling, often used to obtain more reliable estimations from fewer samples. Given a random variable $x \sim p(x)$ and a function $f(x)$. Suppose we want to estimate the expected value of $f(x)$ with sampling:

$$\mu = \mathbb{E}[f(x)] = \int f(x)p(x)dx \approx \frac{1}{n}\sum_{i}^{n} f(x_i), \tag{1}$$

where $\{x_i\}_{i=1,\dots,n}$ are sampled from $p(x)$, and $\frac{1}{n}\sum_{i}^{n} f(x_i)$ is the estimation using the n samples.

Given an *importance distribution* $q(x)$, importance sampling is performed as:

$$\mu = \int f(x)p(x)\frac{q(x)}{q(x)}dx \approx \frac{1}{n}\sum_{i}^{n} f(x_i)\frac{p(x_i)}{q(x_i)}, \tag{2}$$

where $\{x_i\}_{i=1,\dots,n}$ are now sampled from the importance distribution $q(x)$, and $\frac{p(x_i)}{q(x_i)}$, often referred to as the *importance weights*, are used to correct the point-based estimation. If we define $\hat{\mu_q} = \sum_{i}^{n} f(x_i)\frac{p(x_i)}{q(x_i)}$, $\hat{\mu_q}$ is an unbiased estimator of μ.

It has also been shown that the variance of $\hat{\mu_q}$ is,

$$Var[\hat{\mu_q}] = \frac{1}{n}\int \frac{(f(x)p(x) - \mu q(x))^2}{q(x)}dx. \tag{3}$$

Therefore, the optimal importance distribution q^* that offers the lowest variance is:

$$q^*(x) = \frac{|f(x)|p(x)}{\mu} \tag{4}$$

The above equation suggests increasing the sampling probability of x with high absolute function values $|f(x)|$. This theoretical indication supports our core idea of learning to emphasize *critical* events, which lead to hazards and thus large negative values.

B Neural Network Details

Figure 1 demonstrates network architectures of the attention generator and the critic. The attention generator network first concatenates numbers in the current belief b and the observation z into a single vector and feeds it to a feature extractor. The feature extractor consists of 10 fully-connected layers with ReLU activation. The extracted features are input to a Gated Recurrent Unit (GRU) cell [1], which keeps track of history inputs in its latent memory. Based on the latent memory, the network uses two fully-connected layers and a soft-max layer to output the importance distribution q. The critic network has similar architecture. The belief b, observation z, and the generated importance distribution q are first concatenated into a single vector. Then, the vector is fed to a feature extractor containing 6 fully-connected layers with ReLU activation, then input to a GRU cell for tracking memory. Based on the latent memory, the critic network uses another 3 fully-connected layers to finally output a single decimal number, representing the estimated planner value v. See detailed representations of b, z, and q in the following section.

C The POMDP Model for Urban Driving

Our POMDP model for driving in an ill-regulated dense urban traffic is defined as follows:

- **State Modeling**: A world state s encodes:

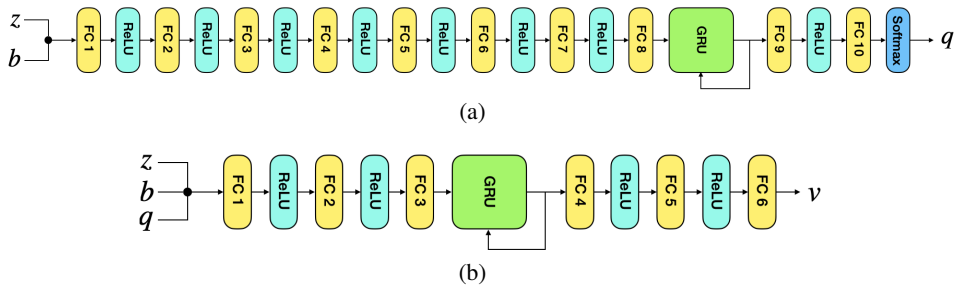

Figure 1: Network architectures of (a) the attention generator and (b) the critic function.

  - the state of the ego-vehicle, $s_c = (p_c, v_c, \alpha_c, P_c)$, where p_c, v_c and α_c denote its position, velocity, and heading direction, and P_c denotes its reference path.
  - observable states of 20 nearest exo-agents, $s_{exo} = \{p_i, v_i, \alpha_i\}_{i=1,\ldots,20}$, where p_i, v_i, α_i are the position, velocity, and heading direction of the ith exo-agent.
  - hidden states of 20 nearest exo-agents, $\theta_{exo} = \{\theta_i\}_{i=1,\ldots,20}$, where θ_i is the intention of the ith exo-agent. Suppose an exo-agent has M potential paths to undertake according to the lane network, the value of its intention θ will be taken from $\{0, \ldots, M-1\}$.

  A belief b is thus a discrete probability distribution defined over the hidden states or intentions of exo-agents, assuming probabilistic independence between different participants. It is represented using $\sum_{i=1}^{20} M_i$ probability values, where M_i is the number of intentions for the ith exo-agent. An importance distribution q is specified in the same way.

- **Action Modeling**: An action a of the ego-vehicle is its acceleration discretized to three values, *ACC*, *CUR*, and *DEC*, meaning to accelerate, keep the current speed, and decelerate. The acceleration and deceleration are $3m/s^2$ and $-3m/s^2$, respectively.
- **Observation Modeling**: An observation z from the environment includes all observable parts of the state s and excludes the hidden intentions. Namely, $z = (s_c, s_{exo})$. Due to perceptual uncertainty, these observations often come with noise. However, in this work, we particularly focused on the uncertainty in human behaviors and ignored perceptual uncertainty, because the latter often has a secondary influence on decision-making.
- **Transition Modeling**: Our transition model assumes the ego-vehicle follows its reference path using a pure-pursuit steering controller and the input acceleration. Exo-agents are not controlled by the algorithm. We assume they take one of their hypothetical intended paths, using the GAMMA motion model [2] to avoid collision with surrounding participants. At each time step, all agents are simulated forward by a fixed duration of $1/3s$. Afterward, small Gaussian noises are added to all transitions to model uncertain human control.
- **Reward Modeling**: The reward function takes into account driving safety, efficiency, and smoothness. When the ego-vehicle collides with any exo-agent, it imposes a severe penalty of $-20 \times (v^2 + 0.5)$ depending on the driving speed v. To encourage driving smoother, we also add a small penalty of -0.1 for the actions *ACC* and *DEC* to penalize excessive speed changes. Finally, to encourage the vehicle to drive at a speed closer to its maximum speed v_{max}, we give it a penalty of $\frac{v - v_{max}}{v_{max}}$ at every time step.

References

[1] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *NIPS 2014 Workshop on Deep Learning, December 2014*, 2014.

[2] Y. Luo, P. Cai, Y. Lee, and D. Hsu. Gamma: A general agent motion model for autonomous driving. *IEEE Robotics and Automation Letters*, 2022.