Interaction-Aware Motion Planning for Autonomous Vehicles via Estimated HMMs of Road User Decisions

Carl-Johan Heiker[1], Yingshuai Quan[1] and Paolo Falcone[1,2]

***Abstract*—In this paper, we propose an interaction-aware predictive motion planning framework for an autonomous vehicle (AV) that must avoid collisions with stochastic and interactive human-driven vehicles (HDVs). We first model the input decision formation in a network of HDV agents as a hidden Markov model (HMM), in which the joint agent input is approximated as a multivariate Gaussian. The distribution parameters are determined by the hidden Markov chain state representing the agent decision configuration in the network. The interaction-aware model predictive controller then determines an optimal AV control input by constrained optimization over the most probable future scenarios, resulting from input ranges specified by the Gaussian parameters of state sequences predicted by a scenario tree. By constraining the AV according to the input bounds of one agent, the controller helps maintain the accuracy of the HMM's joint input predictions. The framework is evaluated in a case study of a traffic intersection that explores how learning different interactive HDV behaviors affects an AV's decision to pass or yield. The proposed interaction-aware controller produced feasible solutions in 93% of simulations, whereas the feasibility of comparative baseline controllers, which do not take interaction into account, was 77%.**

***Index Terms*—Model predictive control, Interaction, Hidden Markov chains, Motion planning, Obstacle avoidance**

I. Introduction

In the transition to fully autonomous transportation systems, new problems arise as autonomous vehicles (AVs) are introduced in traffic scenes that predominantly involve human-driven vehicles (HDVs) [1]. In AV motion planning, predicting HDV trajectories is challenging as human drivers exhibit diverse behaviors and adapt their decisions in response to surrounding vehicles. Bidirectional interaction between the HDVs and the AV has a significant effect on the outcome in traffic scenarios [2], [3]. Therefore, there is a need to predict the HDVs' behaviors conditioned on the AV's actions instead of treating the prediction as an independent process. In this paper, we investigate how a model of interactive decision formation in a traffic scene can be estimated and used to improve predictions in the motion planning strategy of an AV that must avoid collisions with surrounding interactive HDVs.

To achieve this, we require a model of human behavior and interaction. Neural network based methods have been used with reinforcement learning for driver behavior modeling [4], [5], although they are often criticized for their lack of transparency. Alternatively, human road users can be described by explicit models of complex behaviors, such as interaction and decision-making, added on top of basic motion models [6], [7]. In [8], a Markovian model of interactive decision formation among stochastic HDVs was proposed and tested in a traffic intersection example. Based on the model presented in [9] and [10], which was used to describe social networks in [11] and political elections in [12], decision-making agents are modeled as continuous-time Markov chains that interact through transition rate modulation. However, decisions and other abstract signals can not usually be measured directly. Therefore, hidden Markov models (HMMs) that describe underlying but unobservable stochastic processes that have to be inferred from emitted observations are frequently used in traffic applications. HMMs have been used to model, for instance, driver intentions [13], [14], interaction [15] and changes in driver strategies [16], and can be estimated from observations using the Baum-Welch algorithm [17], [18]. Motivated by the flexibility and readability of the HMM-based methods, we model road users as agents with decision configurations represented by the hidden state of an HMM. Each state holds a set of parameters that specify a mode in a multivariate Gaussian distribution of agent inputs. From agent input observations, we then estimate the parameters of the HMM using a Baum-Welch-based algorithm. The estimated HMM describes how agents interactively choose their inputs, and input mode sequence realizations can be predicted using a scenario tree.

Once made available, models of interactive decision formation can be used to improve predictions in the motion planning strategies of AVs, as these are often based on model predictive control (MPC) which determines control input by constrained optimization over a prediction horizon. However, when predictions of surrounding HDVs are uncertain, it is necessary to specify the probability with which specific collision avoidance constraints will be activated. In stochastic MPC, such satisfaction guarantees can be achieved using chance constraints designed to bound the risk of unsafe events [19], [20], [21], [22]. Assumptions on the distribution of the uncertainties, for instance it being multi-modal or Gaussian, has implications for the explicit form of the collision avoidance constraints [23], [24], and on the existence of recursive feasibility guarantees [25]. Alternative to stochastic MPC, the scenario MPC approach instead evaluates a finite set of scenarios representative of the uncertainty distribution [26], [27], [28]. Similarly, [29] generates enough scenarios to guarantee satisfaction of chance constraints by a confidence bound determined by the scenario sample size. While fixed uncertainty distributions can be used

This work is supported by the Swedish Innovation Agency (VINNOVA) under the grant 2018-05005.

[1]Carl-Johan Heiker, Yingshuai Quan and Paolo Falcone are with the Mechatronics group, Departement of Electrical Engineering, Chalmers University of Technology, Gothenburg, Sweden. {heikerc, quany, paolo.falcone}@chalmers.se

[2]Paolo Falcone is with the SAAS - Service d'Automatique et Analyse des Systèmes of the Ecole Polytechnique de Bruxelles at ULB. paolo.falcone@ulb.be

by an AV to predict the decisions of surrounding HDVs that make decisions independently from the actions of the AV itself, such distributions may become invalid if the HDVs also react to the decisions of the AV. In this paper, we introduce an interaction-aware MPC that captures and plans for this bi-directional dependency. The controller uses an estimated HMM of the joint agent input decision process to derive a set of scenarios, formulated as hidden state sequences that specify how the modes in the joint Gaussian input distribution will evolve over a prediction horizon. Using the associated state sequence probabilities, the scenario set can be designed to meet probabilistic safety guarantees. The controller then determines a single control input for the controlled agent by constrained optimization over all scenarios simultaneously. In each prediction stage, scenario specific constraints for both input limitation and collision avoidance are derived from the predicted joint input distribution mode so that the final optimal control input follows the interactive behavior predicted by the HMM.

Our main contributions are i) the estimation of an HMM describing the joint input decision process in a network of agents, and ii) an interaction-aware MPC that uses the HMM to plan the behavior of a controlled agent according to the expected bi-directional interaction with uncontrolled agents.

We formulate our control problem in Section II and describe the HMM in Section III. Section IV treats HMM estimation, while Section V describes our MPC. In Section VI, we prepare the case study that is conducted in Section VII. Finally, we draw conclusions in Section VIII.

II. Problem formulation

A. Controlled agent

In a multi-agent system, we consider a controlled agent with discrete-time dynamics described by

$$x(t_{k+1}) = f\big(x(t_k), u(t_k)\big), \tag{1}$$

where $x(t_k) \in \mathbb{R}^{n_x}$ and $u(t_k) \in \mathbb{R}^{n_u}$ are the state and input vectors at time t_k.

B. Uncontrolled agents

An uncontrolled agent has dynamics described by

$$o(t_{k+1}) = g\big(o(t_k), w(t_k)\big) \tag{2}$$

where the state and input vectors $o(t_k) \in \mathbb{R}^{n_x}$ and $w(t_k) \in \mathbb{R}^{n_u}$ at time t_k have the same domains as x and u.

C. Assumptions

For the two agent types, we make the following key modeling assumptions.

Assumption 1. *The input signal w in (2) is a multi-modal random variable generated by an unknown distribution.*

Assumption 2. *At any time t_k, the most recent input signals $w(t_{k-1})$ from uncontrolled agents are measurable, and the function g in (2) is known.*

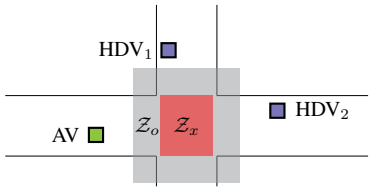


Fig. 1: Intersection example.

Assumption 3. *Each uncontrolled agent tends to avoid entering a perceived collision zone $\mathcal{Z}_o = \{o : o_a \leq o \leq o_b\} \subset \mathbb{R}^{n_x}$, where o_a and o_b are constant vectors, at the same time as other agents. However, this is a behavioral preference rather than a hard constraint.*

Assumption 4. *The controlled and uncontrolled agents influence each other's input decisions bi-directionally, such that $u(t_k)$ and $w(t_k)$ may be correlated.*

D. Motion planning with collision avoidance

Under the assumptions given in Section II-C, the goal is to design an optimal predictive controller that provides an input $u(t_k)$ for a controlled agent such that it reaches a reference state r_x in finite time while avoiding collisions with uncontrolled agents in its perceived risk zone $\mathcal{Z}_x = \{x : x_a \leq x \leq x_b\} \subset \mathbb{R}^{n_x}$, where x_a and x_b are constant vectors. It is possible that $\mathcal{Z}_x \neq \mathcal{Z}_o$. The following example illustrates a traffic scenario in which such a controller is necessary.

Example 1. *In the traffic intersection example depicted in Fig. 1, a controller that determines the acceleration of the AV must predict the future states of itself and the two uncontrolled HDVs in order to prevent collisions between the AV and the HDVs in $\mathcal{Z}_x$. This is challenging for two reasons. First, there is only enough information to predict the actions of the HDVs in probabilities. Second, the AV and the HDVs mutually influence each other's input decisions, which means that a prediction model that treats them as independent will become less accurate in certain parts of the joint input domain.*

To construct an interaction-aware controller that can plan for the effect described in Example 1, we begin by modeling an underlying decision process that will determine how agents select their inputs.

III. Hidden Markov model of agent input decisions

A. Single agent

As in [8], the decision process of a single agent is modeled as a time-homogeneous and ergodic continuous time Markov chain (CTMC) over a set of decision states according to Definition 1.

Definition 1 (Time-homogeneous and ergodic CTMC)**.** *A time-homogeneous CTMC $C(\mathcal{S}, Q, \Pi(t_0))$ describes a state variable $S(t)$ that can transition instantaneously between states in the set $\mathcal{S} = \{s_1, s_2, \ldots, s_M\}$ at any point t in continuous time. The probability to transition from s_i to s_j after a period τ is given by an element $P(i,j) = Pr[S(t_0 +$*

$\tau) = s_j|S(t_0) = s_i]$ *in the* τ*-dependent transition probability matrix* $P(\tau) \in \mathbb{R}^{M\times M}$ *where* $\sum_{j=1}^{M} P(i,j) = 1, \forall i, \tau$ *and* $P(i,j) \geq 0, \forall i, j$*. Time-homogeneity implies that* $P(\tau) = e^{Q\tau}$*, where* $Q \in \mathbb{R}^{M\times M}$ *is a constant matrix of transition rates such that*

$$Q(i,j) = \begin{cases} q_{ij} \geq 0 & \text{if } i \neq j, \\ -\sum_{j=1}^{M} q_{ij} & \text{if } i = j. \end{cases}$$

The vector $\Pi(t) \in \mathbb{R}^{M\times 1}$ *describes the probability that the CTMC is in each state at* $t = t_0 + \tau$ *and is given by solving* $\frac{d}{dt}\Pi(t) = Q^T\Pi(t)$ *from the initial condition* $\Pi(t_0)$*. By definition,* $\sum_{i=1}^{M} \Pi_i(t) = 1$ *and* $\Pi_i(t) \geq 0\ \forall i, t$*. An ergodic CTMC has a nonzero probability to visit all states in finite time, such that* $\lim_{t\to\infty} \Pi(t) = \bar{\Pi}$ *where* $\bar{\Pi}_i > 0,\ \forall i$*.*

To capture the decision influence between several agents, described by Assumption 4, the simultaneous evolution of all agent states can be described in a network model.

B. Agent network

For a set $\mathcal{A}$ of N CTMC agents according to Definition 1, a network state is a tuple $S_X(t) = \langle S_1(t), S_2(t), \ldots, S_N(t)\rangle$ that describes the state variables of all N agents. Hence, it can assume M^N values in the cartesian product $\mathcal{S}_X \in \mathcal{S}_1 \times \mathcal{S}_2, \cdots \times \mathcal{S}_N$.

1) Network CTMC: By definition, the probability that a single CTMC transitions from state s_i to s_j at a singular point in continuous time is zero, which means that the probability that two or more CTMC agents transition at exactly the same time is also zero. Due to this property, the network of CTMC agents can itself be described as a CTMC defined by the network state set $\mathcal{S}_X$, an initial network state probability $\Pi_X(t_0)$ and a transition rate matrix

$$Q_X = \sum_{n=1}^{N} I_{M^{n-1}} \otimes Q_n \otimes I_{M^{N-n}}, \tag{3}$$

constructed from the rate matrices Q_n of the individual agents [9], [8]. In this derivation of Q_X, it is assumed that $\mathcal{S}_n$ is the same for all n, and $\otimes$ denotes the Kronecker product.

2) Uniformization into discrete-time Markov chain formulation: The control application and estimation procedure considered in this paper are described in discrete time, which means that a discrete-time Markov chain (DTMC) network formulation would be sufficiently accurate and less computationally expensive compared to a CTMC.

Definition 2 (Time homogeneous and ergodic DTMC)**.** *A DTMC* $D(\mathcal{S}, P, \Pi(t_0))$ *has a state variable* $S(t_k)$ *that can transition instantaneously between states in the finite set* $\mathcal{S} = \{s_1, s_2, \ldots s_M\}$ *at equidistant points* t_k *in discrete time, where* $k \in \mathbb{N}$*. In a time-homogeneous DTMC, the probability to transition from* s_i *to* s_j *after* k_b *time steps is given by an element* $P^{k_b}(i,j) = Pr[S(t_{k+k_b}) = s_j|S(t_k) = s_i]$ *in the* k_b^{th} *power of the constant one-step transition probability matrix* $P \in \mathbb{R}^{M\times M}$ *where* $\sum_{j=1}^{M} P(i,j) = 1$ *and* $P(i,j) \geq 0$ $\forall i, j$*. The vector* $\Pi(t_k) \in \mathbb{R}^{M\times 1}$ *represents the probability that the DTMC is in each state at* t_k*, and is given by the relationship* $\Pi(t_{k+k_b}) = (P^{k_b})^T\Pi(t_k)$*. By definition,* $\Pi_i(t_k) \geq 0$ *and* $\sum_{i=1}^{M} \Pi_i(t_k) = 1,\ \forall i, k$*. An ergodic DTMC has a nonzero probability to visit all states in finite time, and* $\lim_{k\to\infty} \Pi(t_k) = \bar{\Pi}$ *where* $\bar{\Pi}_i > 0, \forall i$*.*

Generally, a CTMC can be uniformized into a DTMC by transforming its transition rate matrix Q into a transition probability matrix P according to

$$P = \text{unif}(Q, \lambda) = \begin{cases} \frac{q_{ij}}{\lambda} & \text{if } i \neq j, \\ 1 + \frac{q_{ij}}{\lambda} & \text{if } i = j, \end{cases} \tag{4}$$

where $\lambda \geq \max_i -Q(i,i)$ is the uniformization rate. The DTMC has state probabilities $\Pi(t_k)$ that are identical to those of the original CTMC at t_k [30].

3) Simultaneous agent transitions: The transition rate matrix (3) constructed for the CTMC network definition in Section III-B1 does not allow simultaneous transitions in its topology. Because the transition topology is preserved in the uniformization process (4), a DTMC network produced by direct uniformization of the CTMC network transition rate matrix does not allow for simultaneous agent transitions either. However, this is too restrictive for a network model, as stated in the following theorem.

Theorem 1. *A DTMC agent network formulation must permit any number of agents to transition simultaneously.*

Proof. Assume that two CTMCs that follow Definition 1 are observed over the time interval $[t, t+\tau]$ where $\tau > 0$ and t is some point in continuous time. Each CTMC has transition probabilities determined by a τ-dependent matrix of the form $P(\tau) = e^{Q\tau}$. Assume that at some point in time $t \leq t_a \leq t + \tau$, we require that both CTMCs transition. We can approximate $P(t_a)$ by shrinking the interval, but this gives $P(t_a) \approx \lim_{\tau\to 0} P(\tau) = I$ which only allows self loops. Conversely, as long as $\tau > 0$, $P(\tau) \neq I$ which means that any number of transitions other than self loops remain possible for either CTMC on the interval $[t, t+\tau]$. A time-discrete DTMC approximation of the CTMC network that only describes state observations in the interval limits must therefore allow that any number of agents have transitioned between observations. □

4) Network DTMC: To construct a network DTMC $X(\mathcal{S}_X, P_X, \Pi_X(t_k))$ for the agents in $\mathcal{A}$ that agrees with both Definition 2 and Theorem 1, we set

$$P_X = \bigotimes_{n=1}^{N} P_n = \big((P_1 \otimes P_2) \otimes P_3 \ldots\big) \otimes P_N, \tag{5}$$

where the transition probability matrix P_n of each agent a_n is derived through uniformization of its transition rate matrix Q_n according to (4) using a common uniformization rate $\lambda \geq \max_{i,n} -Q_n(i,i)$, corresponding to the highest sum of transition rates out of a state among all agents in the network. With this definition, any number of agents can transition simultaneously with a joint transition probability that is the product of their individual transition probabilities. Additionally, (5) does not require that all agents share the same state set.

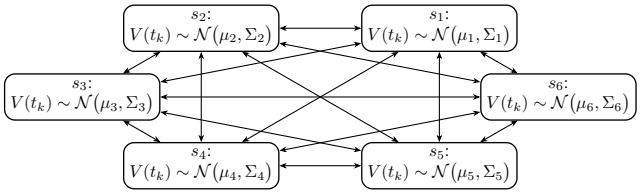


Fig. 2: Network HMM representation of a_1 and a_2.

C. Hidden agent network model

While it may be possible to crudely estimate or guess the parameters of a nominal DTMC describing the isolated decision process of a single agent, the effects of decision influence between several agents are difficult to model. Additionally, decisions in traffic can rarely be measured directly and must be inferred from measurable signals. Therefore, we now assume that the network DTMC $X(\mathcal{S}_X, P_X, \Pi_X(t_k))$ forms the basis of a hidden Markov model of input decision formation in an agent network.

Definition 3 (Hidden Markov model of input decision formation in an agent network)**.** *A hidden Markov model (HMM) $H(X, \theta)$ of input decision formation in an agent network consists of a network DTMC $X(\mathcal{S}_X, P_X, \Pi_X(0))$ that follows Definition 2. However, the value of the network decision state $S_X(t_k)$ cannot be directly observed. Instead, we observe the measurement vector $V(t_k) = \begin{bmatrix} v_1(t_k) & v_2(t_k) & \dots v_N(t_k) \end{bmatrix}^T$, where $v_n(t_k)$ represents the input to agent a_n, which can be controlled or uncontrolled. By Assumption 1, $V(t_k)$ is approximated as a multivariate and multimodal Gaussian random variable $\mathcal{N}(\mu_i, \Sigma_i)$ where the parameter pair $(\mu_i, \Sigma_i) \in \theta$ is determined by the hidden network state $S_X(t_k) = s_i$. $\mu_i \in \mathbb{R}^{Nn_u \times 1}$ is the mean of the network input, while $\Sigma_i \in \mathbb{R}^{Nn_u \times Nn_u}$ is the covariance. For each unobservable state sequence $\mathbb{S}_{1:K} = \big(S_X(t_k) = s_i\big)$, the HMM outputs an observable sequence of observations $\mathbb{V}_{1:K} = \big(V(t_k) = V_k\big)$ where $k \in [1, K] \cap \mathbb{N}$ and V_k denotes a sample from the distribution of $V(t_k)$.*

Definition 3 describes an HMM with a continuous observation density [18]. In the following example, we form a network HMM from two separate agent definitions.

Example 2. *Consider two HMMs that describe the isolated input decision processes of agents a_1 and a_2 with inputs $v_1(t_k) \in \mathbb{R}^{n_u \times 1}$ and $v_2(t_k) \in \mathbb{R}^{n_u \times 1}$. Assume that $v_1(t_k)$ and $v_2(t_k)$ are normally distributed with parameters determined by a state in $\mathcal{S}_1 = \{s_{1,1}, s_{1,2}, s_{1,3}\}$ and $\mathcal{S}_2 = \{s_{2,1}, s_{2,2}\}$. For the HMM agent network representation of a_1 and a_2 in Fig. 2, $V(t_k) = \begin{bmatrix} v_1(t_k) & v_2(t_k) \end{bmatrix}^T \sim \mathcal{N}(\mu_i, \Sigma_i)$ where $\mu_i = \begin{bmatrix} \mu_{1,a} & \mu_{2,b} \end{bmatrix}^T$ and $\Sigma_i = diag(\Sigma_{1,a}, \Sigma_{2,b})$ is determined by a network state $s_i = \langle s_{1,a}, s_{2,b} \rangle \in \mathcal{S}_X = \mathcal{S}_1 \times \mathcal{S}_2$.*

In Example 2, the network HMM simply describes the joint evolution of two independent agents, and each isolated agent model can be reconstructed as a marginal distribution in the network model. In Section IV, we show how the parameters of this nominal network model can be re-estimated to describe effects of decision influence observed in data.

Remark 1. *In e.g. [8], [31], [32], interaction is modeled as social forces that modulate each agent's nominal CTMC transition rates based on the decisions of other agents. While the data-driven approach proposed in this paper produces less general models, the estimated parameters are expected to better reflect the behavior observed in specific scenarios.*

IV. Baum-Welch-based learning algorithm

The Baum-Welch algorithm [17] iteratively updates the parameters of an HMM estimate $\hat{H}(X, \theta)$ until the log likelihood of the parameters with respect to the observations in $\mathbb{V}$ converges to a local, but not necessarily global, maximum. In the following, we outline the key steps of our implementation of the algorithm.

A. Estimation of state transition probabilities

The transition probability matrix P_X of the hidden DTMC X is estimated using the forward/backward procedure [33]. The forward variables $f(i, k)$ (not to be confused with state dynamics in (1)) describe the probability

$$f(i,k) = \Pr[\mathbb{V}_{1:k} | S_X(t_k) = s_i, \hat{H}] \Pr[S_X(t_k) = s_i | \hat{H}]$$

of seeing the observations $\mathbb{V}_{1:k}$ up to t_k and being in state s_i at t_k. This variable is defined recursively as

$$\begin{aligned} f(i,1) &= \Pi_{X,i}(1)\Phi(i,1), \\ f(i,k) &= \Phi(i,k) \sum_j^M f(j,k-1) P_X(j,i), \end{aligned} \tag{6}$$

where $\Pi_{X,i}(1)$ is an arbitrary initialization of the state probability and $\Phi(i, k)$ is an approximation of the probability given by integrating the probability density function of the Gaussian $\mathcal{N}(\mu_i, \Sigma_i)$ over a small region around the observation at t_k, normalized over all i to obtain a probability measure of the case that state s_i generated the observation.

The backward variables describe the probability to make the observations after t_k, given $S_X(t_k) = s_i$, as

$$b(i,k) = \Pr[\mathbb{V}_{k+1:K} | S_X(t_k) = s_i, \hat{H}].$$

Assuming that $S_X(t_k) = s_j$, $b(i, K-1)$ describes every event that can lead to observing $V(t_K)$ given $S_X(t_{K-1}) = s_i$. The backward variables can thus be found recursively as

$$\begin{aligned} b(i,K) &= 1, \\ b(i,k) &= \sum_{j=1}^M b(j,k+1) P_X(i,j) \Phi(j,k+1). \end{aligned} \tag{7}$$

Based on Bayes rule [18], the probability that $S_X(t_k) = s_i$ given the observation sequence $\mathbb{V}_{1:K}$ can be expressed using the forward- and backward variables as

$$\gamma(i,k) = \frac{\Pr[S_X(t_k = s_i), \mathbb{V}_{1:K} | \hat{H}]}{\Pr[\mathbb{V}_{1:K} | \hat{H}]} = \frac{f(i,k) b(i,k)}{\sum_{i=1}^M f(i,k) b(i,k)}. \tag{8}$$

Now, we derive the probability to transition from s_i to s_j given $\mathbb{V}_{1:K}$ and $\hat{H}$ by extending (8) to include the transition from t_k to t_{k+1}. This yields

$$\xi(i,j,k) = \frac{\Pr[S_X(t_k) = s_i, S_X(t_{k+1}) = s_j, \mathbb{V}_{1:K}|\hat{H}]}{\Pr[\mathbb{V}_{1:K}|\hat{H}]} = \frac{f(i,k)P_X(i,j)\Phi(j,k+1)b(j,k+1)}{\sum_{i_1=1}^{M}\sum_{i_2=1}^{M} f(i_1,k)P_X(i_1,i_2)\Phi(i_2,k+1)b(i_2,k+1)}. \tag{9}$$

Note that the denominator of this expression is equal to the denominator of (8). When the previous parameters are computed for D sequences $\mathbb{V}_{d,1:K} \in \mathcal{V}$, we can update the DTMC transition probability matrix estimate as

$$P_X(i,j) = \frac{\sum_{d=1}^{D}\sum_{k=1}^{|\mathbb{V}_{d,1:K}|-1}\xi_d(i,j,k)}{\sum_{d=1}^{D}\sum_{k=1}^{|\mathbb{V}_{d,1:K}|-1}\gamma_d(i,k)}. \tag{10}$$

B. Estimation of Gaussian parameters

The Gaussian parameters in θ can be updated separately by an expectation maximizing step as a weighted sample mean and covariance [18]. In this case, the weight is given by $\gamma(i,k)$, which is the probability of being in state s_i and using the associated parameter pair at time t_k given the sequence $\mathbb{V}_{1:K}$.

However, a single sequence may lack diversity. In this case, only a subset of parameters may give a nonzero value of Φ, causing Σ_i to become ill-conditioned for states s_i that are not represented in the data. While sample diversity can be increased by concatenating observation sequences, this also introduces fictitious transitions in the edges of the data sets, corrupting the estimation of P_X. Hence, we estimate P_X separately for each sequence, but concatenate all γ_d and sequences $\mathbb{V}_{d,1:K}$ before updating the Gaussian parameters according to

$$\mu_i = \frac{\sum_{k=1}^{|\mathbb{V}|}\Gamma(i,k)(\mathbb{V}_k - \mu_i)}{\sum_{k=1}^{|\mathbb{V}|}\Gamma(i,k)}, \tag{11}$$

$$\Sigma_i = \frac{\sum_{k=1}^{|\mathbb{V}|}\Gamma(i,k)(\mathbb{V}_k - \mu_i)(\mathbb{V}_k - \mu_i)^T}{\sum_{k=1}^{|\mathbb{V}|}\Gamma(i,k)}, \tag{12}$$

where $\mathbb{V}$ and Γ are the concatenations of all sequences $\mathbb{V}_{d,1:K} \in \mathcal{V}$ and corresponding γ_d.

C. State pruning

While a large initial state set increases the chances of finding a better local optimum, redundant states do not increase the model likelihood enough to justify the consequent increase in model complexity. This issue can be addressed by removing states s_i from $\mathcal{S}_X$ if $\sum_{k}^{|\mathbb{V}|}\Gamma(i,k) \leq \varepsilon_\gamma$, where ε_γ is a user-defined positive, constant scalar. This can be done as long as $|\mathcal{S}_X| > M_{\min}$, where $M_{\min}$ is the desired minimum number of states. In practice, state pruning is done by removing the Gaussian parameters associated with s_i from θ, deleting the corresponding row and column in P_X and re-normalizing the affected rows.

Algorithm 1 Learning agent network HMMs

procedure NETWORK HMM INITIALIZATION FROM $\mathcal{A}$
  $\mathcal{S}_X = \mathcal{S}_1 \times \mathcal{S}_2 \times \cdots \times \mathcal{S}_N$.
  $P_X = \bigotimes_{n=1}^{N} P_n$ where $P_n = \text{unif}(Q_n, \lambda)$ through (4).
  for network states $s_i = \langle s_{1,a}, s_{2,b}, \ldots s_{N,c}\rangle \in \mathcal{S}_X$ **do**
    $\mu_i = \begin{bmatrix}\mu_{1,a}^T & \mu_{2,b}^T & \cdots & \mu_{N,c}^T\end{bmatrix}^T$.
    $\Sigma_i = \text{blkdiag}(\Sigma_{1,a}, \Sigma_{2,b}, \ldots, \Sigma_{N,c})$.
  end for
end procedure
procedure ESTIMATE NETWORK HMM PARAMETERS
  while $|\bar{L}_{\text{prev}} - \bar{L}| > \varepsilon_\ell$ and $\|P_{X,\text{prev}} - P_X\|_\infty > \varepsilon_P$ **do**
    for sequence $\mathbb{V}_d \in \mathcal{V}$ **do**
      Update f_d by (6), scale by (13).
      Update b_d by (7), scale by (13).
      Update γ_d vars. by (8).
      Update ξ_d vars. by (9).
      $\Gamma \leftarrow \begin{bmatrix}\Gamma & \gamma_d\end{bmatrix}$
      $\mathbb{V} \leftarrow \begin{bmatrix}\mathbb{V} & \mathbb{V}_d\end{bmatrix}$
      $L_d \leftarrow \sum_{k}^{|\mathbb{V}_{d,1:K}|}\log(\ell_d(k))$
    end for
    Compute avg. likelihood $\bar{L} = \frac{1}{|\mathcal{V}|}\sum_{n}^{|\mathcal{V}|} L_n$.
    procedure PRUNE UNUSED STATES
      if $\sum_{k=1}^{|\mathbb{V}|}\Gamma(i,k) < \varepsilon_\gamma$ and $M_{\min} \leq M$ **then**
        $\mathcal{S}_X \leftarrow \mathcal{S}_X \setminus s_i$
      end if
    end procedure
    procedure UPDATE PARAMETER ESTIMATES
      Update P_X by (10).
      for states $s_i \in \mathcal{S}_X$ **do**
        Update μ_i by (11).
        Update Σ_i by (12).
      end for
    end procedure
  end while
end procedure

D. Probability scaling and convergence criterion

To mitigate the numerical underflow resulting from multiplying chains of probabilities in the estimation of P_X, [33] suggests normalizing the forward and backward variables by a scaling factor

$$\ell(k) = \sum_{i=1}^{M} f(i,k), \tag{13}$$

where $f(i,k)$ is the forward variable given by (6). The log likelihood of one sequence $\mathbb{V}_{d,1:K} \in \mathcal{V}$ is then given by $L_d = \sum_{k=1}^{|\mathbb{V}_{d,1:K}|}\log(\ell_d(k))$. We let $\bar{L}_{\text{prev}}$ and $\bar{L}$ denote the average log likelihood of several sequences at the previous and current iteration and use $|\bar{L}_{\text{prev}} - \bar{L}| > \varepsilon_\ell$, where ε_ℓ is a tunable scalar tolerance, as a first convergence criterion for the learning algorithm. The second convergence criterion is $\|P_{X,\text{prev}} - P_X\|_\infty > \varepsilon_P$, where $P_{X,\text{prev}}$ is the estimate from the previous iteration and ε_P is a tunable scalar limit. The steps of the Baum-Welch-based algorithm for estimating an agent network HMM from data are given in Algorithm 1.

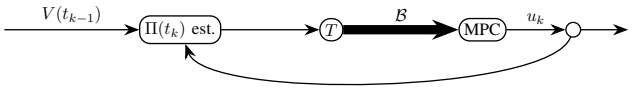


Fig. 3: Interaction-aware MPC procedure.

E. Scenario tree

In scenario-based approaches to MPC, realizations of the uncertainty are derived by sampling its distribution. We use a scenario tree to predict sequences of hidden states from the estimated HMM $\hat{H}(X, \theta)$ over a prediction horizon from t_{k+1} to t_{k+H}.

Definition 4 (Scenario tree)**.** *A scenario tree T constructed for an HMM estimate $\hat{H}(X, \theta)$ that follows Definition 3 is a complete M-ary tree with $N_n = \frac{M^{H+2}-1}{M-1}$ nodes, where $M = |\mathcal{S}_X|$ is the number of network states in X and H is the number of steps in the prediction horizon. Traversal from the root at t_k to one of the $N_b = M^H$ leaves at t_{k+H} produces a unique state sequence $\mathbb{S}_{1:H}$. The associated sequence of state sequence probabilities $\mathbb{P}_{1:H}$ is found through the recursion*

$$\Pi_{X,j}(t_{k+h}) = \Pi_{X,i}(t_{k+h-1}) P_X(i, j), \tag{14}$$

where i and j refers to two consecutive states $S_X(t_{k+h-1}) = s_i$ and $S_X(t_{k+h}) = s_j$ in $\mathbb{S}_{1:H}$. A branch of the scenario tree is defined as the pair $(\mathbb{S}_{1,H}, \mathbb{P}_{1,H})$.

Since the probabilities $\Pi_{X,i}(t_{k+h})$ are products of previous probabilities using a constant P_X and an initial state probability $\Pi_X(t_k)$, they can be re-computed for a new initial state probability $\Pi_X(t_k)$ by substitution in the first step of the recursion.

F. Validation

To asses the prediction quality of a branch $(\mathbb{S}_{1:H}, \mathbb{P}_{1:H})$ generated by T at t_k, we define the quality measure

$$c(i, h) = \Pi_{X,i}(h) \Phi(i, k + h)), \tag{15}$$

for the predicted state $S_X(t_{k+h}) = s_i$. $\Phi(i, k + h))$ is computed using the parameters in state $\mathbb{S}_h$ with respect to a true observation $\mathbb{V}_{k+h}$ in the validation sequence $\mathbb{V}_{k+1:k+H}$. A high score is obtained when the probability $\Pi_{X,i}(h)$ is high, the Gaussian mode specified by s_i has a concentrated covariance Σ_i, and the true observation is close to the mean μ_i.

V. Interaction-aware MPC

The idea behind the interaction-aware MPC, outlined in Fig. 3, is to use a scenario tree T based on a trained HMM to derive a set $\mathcal{B}$ of unique branches that predict how the network of agents will decide their inputs over a finite horizon. All branches will be evaluated simultaneously in the controller, which selects a final input $u(t_k)$ to a controlled agent in the network. For simplicity, we assume that only one agent is controlled.

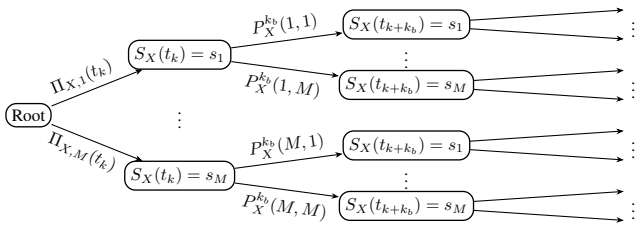


Fig. 4: Scenario tree with reduced prediction resolution.

A. Controller design

1) Branch set derivation: First, a scenario tree T that follows Definition 4 predicts a set $\mathcal{B}$ of the most probable unique branches over the controller's prediction horizon. To reduce computational complexity, we utilize a technique similar to the ones used in e.g. [22], [34] in which sequence probabilities are evaluated at every k_b^{th} step of the horizon by replacing P_X with $P_X^{k_b}$ in (14), such that $\Pi_{X,j}(t_{k+k_b}) = \Pi_{X,i}(t_k) P^{k_b}(i, j)$. The result is a state sequence with a lower resolution than the prediction horizon, which requires us to approximate the value of the state and probability chain between steps. For processes with slowly varying state transitions, it is sufficient to use zero-order hold to extend the sequences in these periods. The scenario tree with this approximation enabled is shown in Fig. 4.

Remark 2. *The number of branches to evaluate can either be selected by thresholding their cumulative probability to be no less than a user-defined probability $\varepsilon \in (0, 1]$, or constant. While the former alternative can be used to assert that safety constraints are satisfied with a certain probability, much like chance constraints, the latter approach is used in the simulations conducted in this paper for simplicity.*

2) Decision variables: At each prediction stage h of each branch $(\mathbb{S}_{b,1:H}, \mathbb{P}_{b,1:H}) \in \mathcal{B}$, the controller determines continuous decision variables $u_{b,h} \in \mathbb{R}^{n_u}$ and $x_{b,h} \in \mathbb{R}^{n_x}$ representing the input and state of (1) at t_{k+h}. Note that $u_{b,1}$ represents the input applied at t_k while $x_{b,1}$ is the resulting state at t_{k+1}. The slack variables $\rho_{b,l,h} \in \mathbb{R}^{n_u}$ and $\rho_{b,u,h} \in \mathbb{R}^{n_u}$ are used to relax the upper and lower input bounds if necessary to maintain feasibility. The single input that the controller will apply at t_k is described as the continuous decision variable $u_k \in \mathbb{R}^{n_u}$.

Subsets of branches $\mathcal{B}_g \subseteq \mathcal{B}$ that share the same initial state but diverge later must be included or excluded simultaneously. If the single initial input u_k obeys the constraint of that state, all branches in $\mathcal{B}_g$ may be realized, whereas none of them will be realized if u_k does not follow the constraints. The binary decision variable $z_{c,g} \in \{0, 1\}$ encodes this choice, where $z_{c,g} = 1$ indicates that all branches in $\mathcal{B}_g$ are included in the optimization. Finally, another binary decision variable $z_{o,b,n,m} \in \{0, 1\}$ activates the m^{th} collision avoidance constraint for the n^{th} uncontrolled agent in the branch with index b.

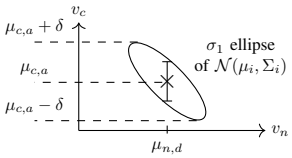


Fig. 5: The controlled agent a_c's input decision variable is constrained to lie on the interval $[\mu_{c,a}-\delta, \mu_{c,a}+\delta]$ based on the predicted skewed agent network input distribution $\mathcal{N}(\mu_i, \Sigma_i)$.

3) Objective function: The objective function that will be minimized by the interaction-aware MPC is

$$J = \sum_{\mathcal{B}_g \subseteq \mathcal{B}} z_{c,g} \sum_{b=1}^{|\mathcal{B}_g|} \frac{1}{|\mathcal{B}_g|} \sum_{h=1}^{H} (c_d - \mathbb{P}_{b,h}) d_{b,h} C d_{b,h}^T. \tag{16}$$

The vector $d_{b,h} = \begin{bmatrix} x_{b,h} - r_{b,h} & u_{b,h} & \rho_{b,l,h} + \rho_{b,u,h} \end{bmatrix}^T$ contains the state variable's divergence from the reference $r_{b,h}$, the input, and the sum of slack variables, while $C = \text{blkdiag}(q, r, c)$ holds the corresponding cost matrices q, r and c so that $d_{b,h} C d_{b,h}^T$ forms the quadratic cost of scenario b at stage h. The probability discount $c_d - \mathbb{P}_{b,h}$, where $c_d > 1$ and $\mathbb{P}_{b,h}$ is the probability of branch b at stage h, reduces the cost of prioritizing a branch with high probability. The branch subset choice variable $z_{c,g}$ determines whether or not the branch subset $\mathcal{B}_g$ should be realized and thereby contribute to the objective function. The cost is scaled by $1/|\mathcal{B}_g|$ to compare subsets of different sizes equally.

4) State constraints: For all b and h, the state variable is bounded according to

$$x_{\min} \leq x_{b,h} \leq x_{\max} \tag{17}$$

and follows the state dynamics (1) by

$$x_{b,h+1} = f(x_{b,h}, u_{b,h}). \tag{18}$$

5) Input constraints: A branch with index b predicts that the agent network's joint input vector $V(t_{k+h}) = \begin{bmatrix} v_1(t_{k+h}) & v_2(t_{k+h}) & \dots & v_N(t_{k+h}) \end{bmatrix}^\top$ will be normally distributed with the parameters μ_i and Σ_i of state $S(t_{k+h}) = s_i$ at stage h. The predicted input $v_c(t_{k+h})$ of a controlled agent $a_c \in \mathcal{A}$ is represented by a subset of the elements in $V(t_{k+h})$. Therefore, its input decision variable $u_{b,h}$ should lie within a confidence interval specified by the conditional distribution $v_c(t_{k+h})|v_{n\neq c}(t_{k+h})$, where $n \neq c$ denotes the indices of the remaining agents in the network. Assume that the network is predicted to be in $s_i \in \mathcal{S}_X$ where the controlled agent is in $s_{c,a} \in \mathcal{S}_c$. If there is no interaction, agent inputs are mutually independent and $v_c(t_{k+h}) \sim \mathcal{N}(\mu_{c,a}, \Sigma_{c,a})$, where $\mu_{c,a}$ and $\Sigma_{c,a}$ are the individual mean and covariance of the controlled agent. To constrain the agent input decision variable $u_{b,h}$ to lie within one standard deviation of $\mathcal{N}(\mu_{c,a}, \Sigma_{c,a})$, we may set $\mu_{c,a} - \sigma_{c,1} \leq u_{b,h} \leq \mu_{c,a} + \sigma_{c,1}$, where $\sigma_{c,1}$ denotes one standard deviation. However, agent interaction will introduce cross terms in Σ_i and increase the skewness of $\mathcal{N}(\mu_i, \Sigma_i)$, as illustrated in Fig. 5. To account for the skewed covariance, we instead set $\mu_{c,a} - \delta \leq u_{b,h} \leq \mu_{c,a} + \delta$, where δ is the projection of the σ_1 level curve of $\mathcal{N}(\mu_i, \Sigma_i)$ onto the axis of v_c. If agents are independent, this method is equivalent to the previously mentioned way of setting the constraint according to the individual agent distribution.

In total, we set the input constraint

$$\mu_{c,a} - \delta - \rho_{b,l,h} \leq u_{b,h} \leq \rho_{b,u,h} + \mu_{c,a} + \delta, \tag{19}$$

which includes the slack variables $\rho_{b,l,h}$ and $\rho_{b,u,h}$ which may relax the lower and upper HMM-based input bounds, respectively, to maintain feasibility. The slack variables themselves are upper bounded according to

$$\rho_{b,l,h} \leq |u_{\min} - (\mu_{c,a} - \delta)| \tag{20a}$$

$$\rho_{b,u,h} \leq |u_{\max} - (\mu_{c,a} + \delta)| \tag{20b}$$

so that $u_{b,h}$ is bounded by the same constant for all values of $\mu_{c,a}$ and δ.

6) Branch subset constraints: For all b such that $(\mathbb{S}_{b,1:H}, \mathbb{P}_{b,1:H}) \in \mathcal{B}_g$, $\mathbb{S}_{b,1} = s_i$ for exactly one $s_i \in \mathcal{S}_X$. Hence, the initial input in these branches should be the same, which is achieved by enforcing the constraint

$$-(1 - z_{c,g})\mathbf{M} \leq u_k - u_{b,1} \leq (1 - z_{c,g})\mathbf{M} \tag{21}$$

When $z_{c,g} = 1$, the initial input of all branches in $\mathcal{B}_g$ will be forced to equal the single input choice variable u_k, meaning that they will be equal to each other. When $z_{c,g} = 0$, the bounds in (21) are relaxed by a sufficiently large constant scalar $\mathbf{M}$ following the Big-M method in [35]. Since the initial input cannot satisfy the constraints of several subsets $\mathcal{B}_g$ simultaneously, we impose

$$\sum_{g=1}^{G} z_{c,g} = 1, \tag{22}$$

where G is the number of branch subsets.

7) Collision avoidance constraints: For each branch in $\mathcal{B}$, we can define the controlled agent's reachable set of states at t_{k+h+1} as $\mathcal{X}_{b,h+1}$ through

$$\mathcal{X}_{b,h+1} = \{f(x_{b,h}, u_{b,h}) | x_{b,h} \in \mathcal{X}_{b,h}, u_{b,h} \in \mathcal{U}_{b,h}\}, \tag{23}$$

where $\mathcal{X}_{b,h}$ and $\mathcal{U}_{b,h}$ are the sets of reachable states and allowed inputs at t_{k+h}. Analogously, (23) yields the relationship between the reachable sets $\mathcal{O}_{b,h+1}$ and $\mathcal{O}_{b,h}$ for an uncontrolled agent by substituting the dynamics for (2) and $\mathcal{U}_{b,h}$ for the uncontrolled agent's set of allowed inputs $\mathcal{W}_{b,h}$.

Definition 5. *At prediction stage h, a reachable controlled agent subset $\mathcal{X}_{b,h}^{safe} \subset \mathcal{X}_{b,h}$ is labeled safe with respect to the uncontrolled agent a_n iff $\mathcal{X}_{b,h}^{safe} \cap \mathcal{O}_{b,n,h} = \emptyset$.*

To enforce $x_{b,h} \in \mathcal{X}_{b,h}^{\text{safe}}$ with respect to $\mathcal{O}_{b,n,h}$, we define M_o constraints of the form

$$A_m x_{b,h} \leq b_m + (1 - z_{o,b,n,m})\mathbf{M}, \tag{24a}$$

$$\sum_{m=1}^{M_o} z_{o,b,n,m} = 1, \tag{24b}$$

where (24a) constrains the state of the controlled agent to be in a half space that does not contain $\mathcal{O}_{b,n,h}$. The binary variable $z_{o,b,n,m}$ determines whether the constraint (24a) is activated,

following the Big-M method, and (24b) imposes that exactly one of the M_o constraints is selected. In the following theorem, we show how many constraints of the form (24a) are required to avoid a set of N uncontrolled agents.

Theorem 2. *A reachable subset $\mathcal{X}_{b,h}^{safe} \subset \mathcal{X}_{b,h}$ of the domain $\mathcal{D}$ that is safe with respect to the possibly non-convex closed reachable sets $\mathcal{O}_{b,n,h}$ of uncontrolled agents $a_n \in \mathcal{A}$ is given by $\mathcal{X}_{b,h}^{safe} = \mathcal{X}_{b,h} \setminus \bigcup_{n=1}^{N} \mathcal{O}_{b,n,h}$. This set, which is generally non-convex, can be approximated using M_o integer constraints of the form* (24a) *for each set $\mathcal{O}_{b,n,h}$. At most $(M_o)^N$ constraint candidates have to be evaluated at each prediction stage h in each branch to find convex subset of $\mathcal{X}_{b,h}^{safe}$ for the controlled agent.*

Proof. At prediction stage h of a branch, we can always separate the domain $\mathcal{D}$ into one half space that contains $\mathcal{O}_{b,n,h}$ and one that is empty, using the integer constraint (24a). Hence, any specific choice of $z_{o,b,n,m}$ will define a safe half-space $\mathcal{H}_{b,n,h}^{\text{safe}}$ that is open in a direction that points away from $\mathcal{O}_{b,n,h}$. For each of the N uncontrolled agents, we must choose one of M_o constraints that ensures that the controlled agent stays in the half space that is safe with respect to $\mathcal{O}_{b,n,h}$. However, the intersection of all such half spaces $\mathcal{H}_{b,N,h} = \bigcap_{n=1}^{N} \mathcal{H}_{b,n,h}^{\text{safe}}$ may be empty for certain values of $z_{o,m}$ and n, or it does not intersect with the controlled agents' reachable set $\mathcal{X}_{b,h}$. As there are $(M_o)^N$ ways of combining the constraints, we must evaluate at most $(M_o)^N$ safe convex sets to find if they are both non-empty and intersect the reachable set $\mathcal{X}_{b,h}$. □

Remark 3. *Although leaving $z_{o,b,n,m}$ as a decision variable yields a globally optimal constraint selection, Theorem 2 shows that this requires evaluating up to $(M_o)^N$ constraint combinations in each prediction stage of each branch for N agents. When a reliable heuristic for selecting $z_{o,b,n,m}$ exists, it can significantly reduce computational costs with little loss in solution quality.*

B. Optimization problem

Given an HMM that follows Definition 3, the proposed interaction-aware MPC solves the optimization problem

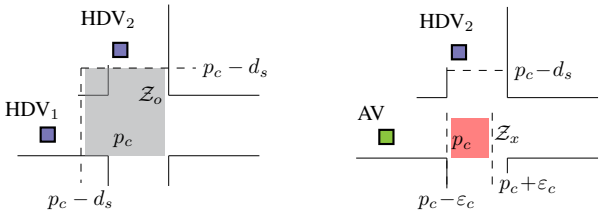
$$\underset{\substack{d_{b,h},u_k,\\ z_{c,g},z_{o,b,n,m}}}{\text{minimize}} \sum_{\mathcal{B}_g \subseteq \mathcal{B}} z_{c,g} \sum_{b=1}^{|\mathcal{B}_g|} \frac{1}{|\mathcal{B}_g|} \sum_{h=1}^{H} (c_d - \mathbb{P}_{b,h}) d_{b,h} C d_{b,h}^T$$

$$\text{s.t. } (17), (18), (19), (20a), (20b), (21), (22), \quad (24a) \text{ and } (24b), \quad \forall g, b, n, h, m. \tag{25}$$

At each step t_k, the controller minimizes the objective function (16) with respect to the vector $d_{b,h}$ used to formulate the stage cost, the final input u_k and the binary variables $z_{c,g}$ and $z_{o,b,n,m}$ for choosing input branch and setting collision constraints. The minimization in (25) is subject to state constraints (17) and (18), slack variable constraints (20a) and (20b), and branch subset choice constraints (21) and (22). Additionally, input constraints (19) and collision avoidance constraints (24a) and (24b) are predicted using the HMM description of interactive agent network input decision formation, and imposed to produce an interaction-aware control strategy.

Algorithm 2 Interaction-aware MPC subroutine at t_k

procedure PREDICT BRANCHES
  Initial state probs. $\Pi_{X,i}(t_k) \leftarrow \Phi(i, k-1)$, $\forall i$.
  Get branch set $\mathcal{B}$ from tree eval. of $\hat{H}(P_X, \theta)$.
end procedure
procedure CONSTRUCT OPTIMIZATION PROBLEM
  Set branch subset constr. (22).
  Set subset initial input equality constr. (21).
  for $b = 1, 2, \ldots, |\mathcal{B}|$ **do**
    Add branch cost to (16).
    Compute controlled agent reachable sets $\mathcal{X}_{b,h}$.
    for $n = 1, 2, \ldots, N$ **do**
      Compute reachable sets $\mathcal{O}_{b,n,h}$ $\forall h$.
    end for
    for each prediction stage h **do**
      Set system constr. (17), (18).
      Set input and slack constr. (19), (20a), (20b).
      Set collision avoidance constr. (24a), (24b).
    end for
  end for
end procedure
Solve integer optimization problem (25) and obtain u_k.



(a) Learning problem. (b) Control problem.

Fig. 6: Traffic intersection scenario used in simulations.

VI. SIMULATION SETUP

A. Traffic intersection scene

We now return to the traffic intersection presented in Example 1 and consider situations in which exactly two vehicles approach the intersection in straight lines from west to east, and north to south, respectively, and address two problems.

1) Learning problem: In Fig. 6a, the scene contains HDV_1 and HDV_2. Using data from several instances of this process, we will estimate an agent network HMM $\hat{H}(X, \theta)$ of how the HDVs choose their inputs interactively.

2) Control problem: In Fig. 6b, we replace HDV_1 with a controlled AV that uses the interaction-aware MPC presented in Section V to reach its state reference while avoiding collisions with HDV_2 in $\mathcal{Z}_x$.

Remark 4. *For traffic scenarios in which the number of road users varies over time, a similar agent-based modeling framework could be used to describe an average road user behavior.*

B. HDV model

1) Dynamics: The state vector of HDV_n, where $n = 1, 2$, is $o_n(t_k) = \begin{bmatrix} p_n(t_k) & \nu_n(t_k) \end{bmatrix}^T$ which describes its position and speed along a straight path. We use a linear double integrator model, discretized with the time step $t_s = 0.02$, to describe the development of $o_n(t_k)$ as

$$o_n(t_{k+1}) = Ao_n(t_k) + Bw_n(t_k), \tag{26}$$

where

$$A = \begin{bmatrix} 1 & 1.02 \\ 0 & 1 \end{bmatrix}, \quad B = 2 \cdot \begin{bmatrix} 10^{-3} \\ 10^{-2} \end{bmatrix}. \tag{27}$$

In each simulation, the initial state is chosen according to

$$o_n(t_0) = \begin{bmatrix} 10 + d_{p,0} & 5 + d_{\nu,0} \end{bmatrix}^T \tag{28}$$

where $d_{p,0} \sim \mathcal{N}(0, 0.1)$ and $d_{\nu,0} \sim \mathcal{N}(0, 0.1)$.

2) Behavior model: Each HDV considers a risk zone $\mathcal{Z}_o = \{p_n : |p_c - p_n| \leq d_s\}$, shown in 6a, where $p_c = 20$ is the intersection center and $d_s = 4$ is a safety distance. HDV_n integrates the current speed of itself and other vehicles to predict if they will be in $\mathcal{Z}_o$ simultaneously. If so, it will choose a speed reference $r_{\nu,n}(t_k)$ such that

$$r_{\nu,n}(t_k) = \begin{cases} 1.3r_\nu & \text{if } b_n(t_k) = a \\ 0 & \text{if } b_n(t_k) = p \end{cases}$$

where $b_n(t_k)$ is a behavior state variable that can be a or p, representing aggressive or passive, and r_ν is a nominal speed reference. A simple two-state DTMC with transition probability matrix

$$P_n = \begin{bmatrix} 1 - p_{ap} & p_{ap} \\ p_{pa} & 1 - p_{pa} \end{bmatrix} \tag{29}$$

describes the evolution of $b_n(t_k)$. In (29), p_{ap} and p_{pa} is the probability to transition from a to p and vice versa. Each HDV has an initial behavior state and a fixed matrix P_n.

3) Reference tracking: Each HDV follows its piecewise constant reference $r_{\nu,n}(t_k)$ using a PID controller. The input to (26) is, for notational clarity, described in $\mathcal{Z}$-domain as $W_n(z) = G(z)\Big(F(z)\big(r_{\nu,n} - \nu_n(z) - d_\nu(z)\big) + d_w(z)\Big)$ where $G(z) = \frac{1}{3 - z - z^2}$ produces a moving average of the three most recent inputs, $F(z) = \frac{10.63z^2 - 20.48z + 9.87}{z^2 - 1.45z + 0.45}$ is a PID controller and $d_\nu(z) \sim \mathcal{N}(0, 10^{-3})$ and $d_w(z) \sim \mathcal{N}(0, 1)$ are the measurement and input disturbances.

C. AV model

The controlled AV follows the same discretized double integrator as the HDVs, such that its position and speed $x(t_k) = \begin{bmatrix} p(t_k) & \nu(t_k) \end{bmatrix}^T$ develops according to

$$x(t_{k+1}) = Ax(t_k) + Bu(t_k) \tag{30}$$

where A and B are described in (27). The initial state of the AV in all simulations is generated in the same way as the initial condition for the HDVs (28). For simplicity, the AV considers a slightly widened stopping line $\mathcal{Z}_x = \{p : p_c - \varepsilon_c \leq p \leq p_c + \varepsilon_c\}$ where $\varepsilon = 0.2$ as the collision risk zone. This zone is depicted in Fig. 6b.

Algorithm 3 Intersection scenario simulation

```
procedure SIMULATE VEHICLES
    while t < T do
        for each vehicle with index n do
            if Vehicle is HDV_n then
                Predict own pos. p_n(t_{k+h}).
                Predict others' pos. p_m(t_{k+h}), m ≠ n.
                if p_n(t_{k+h}), p_m(t_{k+h}) ∈ Z_o then
                    Sample b_n(t_{k+1}) from p_n.
                end if
                Set r(t_k) according to b_n(t_k).
                Derive w_n(t_k), update (26).
            else
                Derive u(t_k) by Alg. 2, update (30).
            end if
        end for
        V_{k:k} ← V(t_k)
    end while
end procedure
```

	s_1	s_2	s_3	s_4	s_5	s_6	s_7	s_8	s_9
$\mu_{1,a}$	-4	-4	-4	0	0	0	4	4	4
$\mu_{2,b}$	-4	0	4	-4	0	4	-4	0	4
Σ_i	I_2	I_2	I_2	I_2	I_2	I_2	I_2	I_2	I_2

TABLE I: Gaussian parameters $\mu_i = \begin{bmatrix} \mu_{1,a} & \mu_{2,b} \end{bmatrix}^T$ and Σ_i of each state in $\hat{H}_1$.

D. Learning and validation

1) Data generation: In each experiment, 700 network input observation sequences for training and 300 for validation are generated by simulating the scene in Fig. 6a using Algorithm 3 with HDV models described in Section VI-B.

2) Learning algorithm: To create an initial estimate of $\hat{H}(X, \theta)$, which we denote $\hat{H}_1$, we define two agents with identical state sets $\mathcal{S}_n = \{s_{n,1}, s_{n,2}, s_{n,3}\}$ for $n = \{1, 2\}$. After uniformization, we obtain transition probability matrices

$$P_n = \begin{bmatrix} 0.450 & 0.250 & 0.300 \\ 0.350 & 0.250 & 0.400 \\ 0.450 & 0.500 & 0.050 \end{bmatrix}.$$

The state sets map to the set of individual means $\{-4, 0, 4\}$ and variances $\{1, 1, 1\}$ for the individual normal distributions of the scalar $v_n(t_k)$. The network state set is $\mathcal{S}_X = \mathcal{S}_1 \times \mathcal{S}_2$ and the associated network's Gaussian parameters are found in Table I. We estimate a new model using the Baum-Welch algorithm from Section IV using the derived training sequences. The maximum number of iterations for Baum-Welch is 1000, and the pruning limit is $\varepsilon_\gamma = 0.2$ with a minimum number of $M_{\text{min}} = 4$ states.

3) Validation: We validate an estimated HMM by repeating the procedure in Section IV-F 1000 times starting from randomly chosen time points t_k to form an average validation score at each step h and state s_i. The final score $c(h)$ is the average of $c(i, h)$ over all states.

E. Proposed interaction-aware MPC

1) Tuning: For simplicity, we fix the number of predicted branches to $|\mathcal{B}| = 5$. The branching step is $k_b = 7$ and we choose the MPC horizon $H = 55$. The branch cost matrices in (25) are set to $q = \text{diag}(\begin{bmatrix}1000 & 1000\end{bmatrix})$, $r = 1$ and $c_\rho = 1000$. The state reference is $r_{b,h} = \begin{bmatrix}1500 & 15\end{bmatrix}^T$ in all branches and experiments. The upper and lower bounds for the state variable in constraint (17) are $x_{\min} = \begin{bmatrix}-100 & 0\end{bmatrix}^T$ and $x_{\max} = \begin{bmatrix}1000 & 30\end{bmatrix}^T$ in all branches. The state dynamics constraint (18) is formulated for the linear AV model (30). We construct input constraints (19) related to the σ_1 curve of the network input distribution. The lower slack variable bound (20a) is set using $u_{\min} = -7\ [m/s^2]$ so that emergency braking is always possible, but the upper slack variable (20b) is set to zero to avoid aggressive behavior.

In this example, collisions are only possible at stages h in which the predicted reachable positions of HDV_2 intersect $\mathcal{Z}_x$. Hence, (24a) reduces to

$$p_c + \varepsilon_c - z_b \mathbf{M} \leq x_{b,h} \leq p_c - \varepsilon_c + (1 - z_b)\mathbf{M}, \qquad (31)$$

which describes either stopping before or having passed $\mathcal{Z}_x$ by h. Since the zone is narrow, we can reduce computational complexity further by setting the constraint for the few h in which collisions are predicted using a single variable z_b in each branch.

2) Version with heuristic collision avoidance: We also formulate a version of the proposed MPC identical to the one described in Section VI-E1 but with heuristic rule-based collision avoidance constraints as suggested in Remark 3. We evaluate (31) for both values of z_b and always choose the passing option if it is feasible.

F. Baseline MPC

We formulate three baseline MPCs that each solves a version of the optimization problem

$$\underset{x,u,z,\rho_l}{\text{minimize}} \sum_{h=1}^{H} (x_h - r_h) q (x_h - r_h)^T + r u_h^2 + c_\rho \rho_{l,h}^2 \quad (32a)$$

$$\text{s.t. } x_{\min} \leq x_h \leq x_{\max}, \quad (32b)$$

$$x_{h+1} = f(x_h, u_h), \quad (32c)$$

$$a_u - \rho_{l,h} \leq u_h \leq b_u \quad (32d)$$

$$\rho_{l,h} \leq |u_a - u_{\min}| \quad (32e)$$

$$p_c + \varepsilon_c - zM \leq x_h \leq p_c - \varepsilon_c + (1 - z)M \quad (32f)$$

A baseline MPC does not receive predictions from an HMM and therefore minimizes the total stage cost, formulated as in (16), of a single trajectory. The state constraints (32b) and (32c) and the collision avoidance constraint (32f) are identical to those in (25). However, a baseline MPC implements a rule-based driving style through its input constraint (32d) and in how it assumes the speed range of surrounding HDVs, shown in Table II. The baseline B_2 is passive with a lower acceleration range and predicts HDV speed $\hat{\nu}(t_{k+h})$ to be higher than the current observation. B_3 is aggressive and uses an opposing strategy while B_1 can be either passive or aggressive. Other parameters in the baseline controllers have the same values as the proposed controller.

	Predicted HDV speed range	Bounds in (32d)
B_1	$\nu_n(t_k) - 1 \leq \hat{\nu}_n(t_{k+h}) \leq \nu_n(t_k) + 1$	$a_u = -4$, $b_u = 4$
B_2	$\nu_n(t_k) \leq \hat{\nu}_n(t_{k+h}) \leq \nu_n(t_k) + 3$	$a_u = -4$, $b_u = 1$
B_3	$\nu_n(t_k) - 3 \leq \hat{\nu}_n(t_{k+h}) \leq \nu_n(t_k)$	$a_u = 1$, $b_u = 4$

TABLE II: Baseline MPC predicted HDV input and own input range.

VII. Case study

In three different experiments, we first solve the learning problem described in Section VI-A1 by producing two HMM estimates $\hat{H}_2$ and $\hat{H}_3$ from the same initial estimate $\hat{H}_1$ given in Section VI-D2. $\hat{H}_2$ has the same number of states as $\hat{H}_1$, whereas $\hat{H}_3$ has a reduced number of states as a result of state pruning, discussed in Section IV-C. All three models are validated using the measure presented in Section VI-D3 computed with transient, stationary and uniform state probability predictions.

Next, the control problem is solved by replacing HDV_1 with the controlled AV to evaluate how an interaction-aware MPC I that uses $\hat{H}_3$ compares to the baseline MPCs in Section VI-F and to I_h, a version of I with the heuristic collision avoidance constraints discussed in Section VI-E2. In each experiment, we simulate the intersection scenario 100 times for each controller using Casadi [36] with the Bonmin solver [37] in Matlab.

A. HDV_1 aggressive and HDV_2 passive

In the first case, the behavior state probabilities are set to $p_{i,a} = 0.9$ and $p_{i,p} = 0.1$, where $i = \{a, p\}$ for HDV_1 and $p_{i,a} = 0.1$ and $p_{i,p} = 0.9$ for HDV_2. The effect is shown in the scatter plots in Figs. 7a - 7c, where HDV_1 often accelerates while HDV_2 brakes.

1) Learning results: The large spread in the mode placement of the initial untrained estimate $\hat{H}_1$, seen in Fig. 7a gives a low validation score in 7g. The trained estimate $\hat{H}_2$ and the trained, pruned version $\hat{H}_3$ show significant improvement in Figures 7b and 7c. Gaussians with higher covariance cover less populated regions, while areas around $V(t_k) = \begin{bmatrix}4 & -4\end{bmatrix}^T$ are represented by narrower Gaussians. In Fig. 7c, s_3 is placed exactly in this value and emerges as the most probable state in Fig. 7f.

The validation scores of $\hat{H}_2$ and $\hat{H}_3$, shown in Figures 7h and 7i, are initially higher for transient probabilities due to accurate estimation of the initial state, but quickly drops to the performance of stationary probabilities. This is because $\hat{H}_2$ and $\hat{H}_3$ accurately capture that states around $V(t_k) = \begin{bmatrix}4 & -4\end{bmatrix}^T$ are more probable over all, whereas the time at which the sudden HDV behavior changes will occur is difficult to predict specifically. For a system that stays long periods in each state without such fluctuations, we would see a longer transient in the score. Despite this, using estimated DTMC probabilities is better than guessing between modes with uniform probabilities. Importantly, the similarity between the validation scores of $\hat{H}_2$ and $\hat{H}_3$ reveal the state redundancy of $\hat{H}_2$, and therefore $\hat{H}_3$ is used in the controller I.

2) Controller simulation: All controllers are feasible in all 100 simulations. Over all, a successful strategy for all controllers is to choose its maximal allowed input, as HDV_2

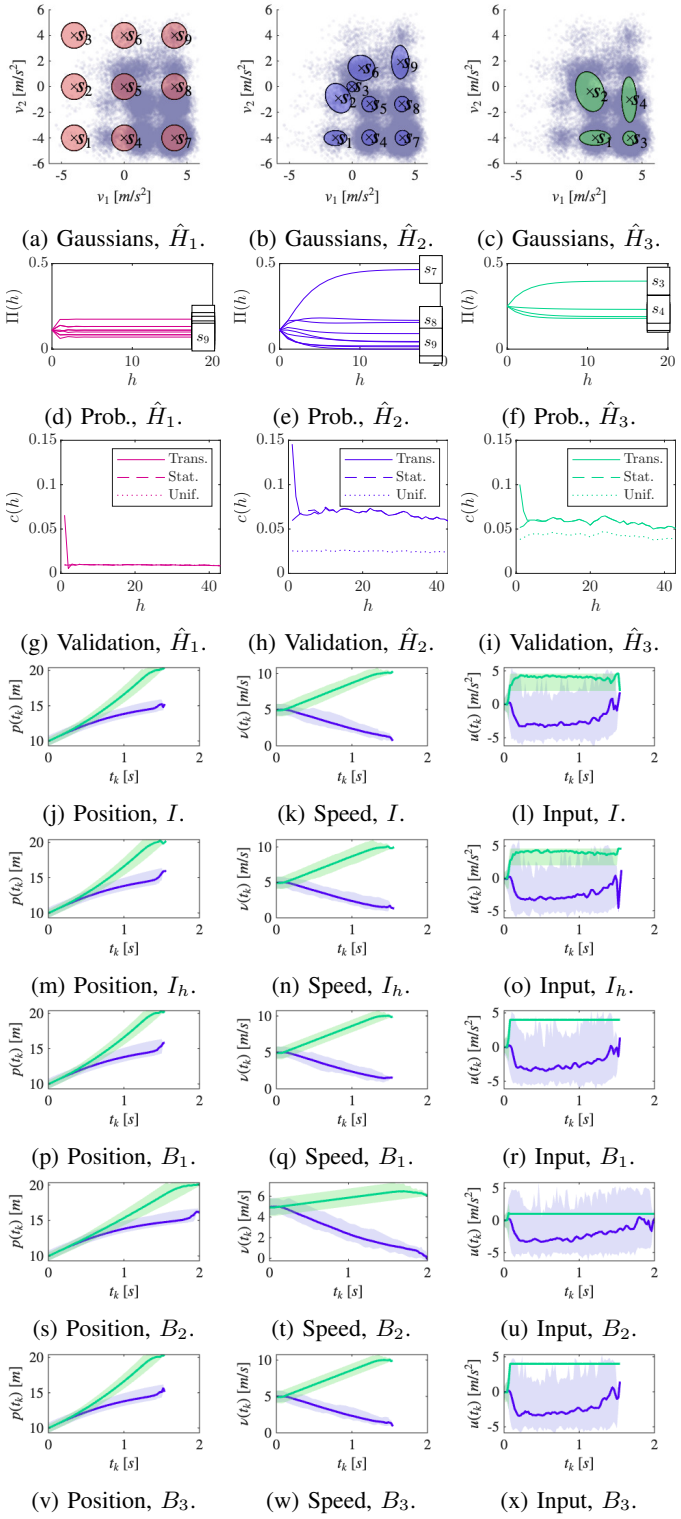


Fig. 7: Results, HDV_1 aggressive and HDV_2 passive.

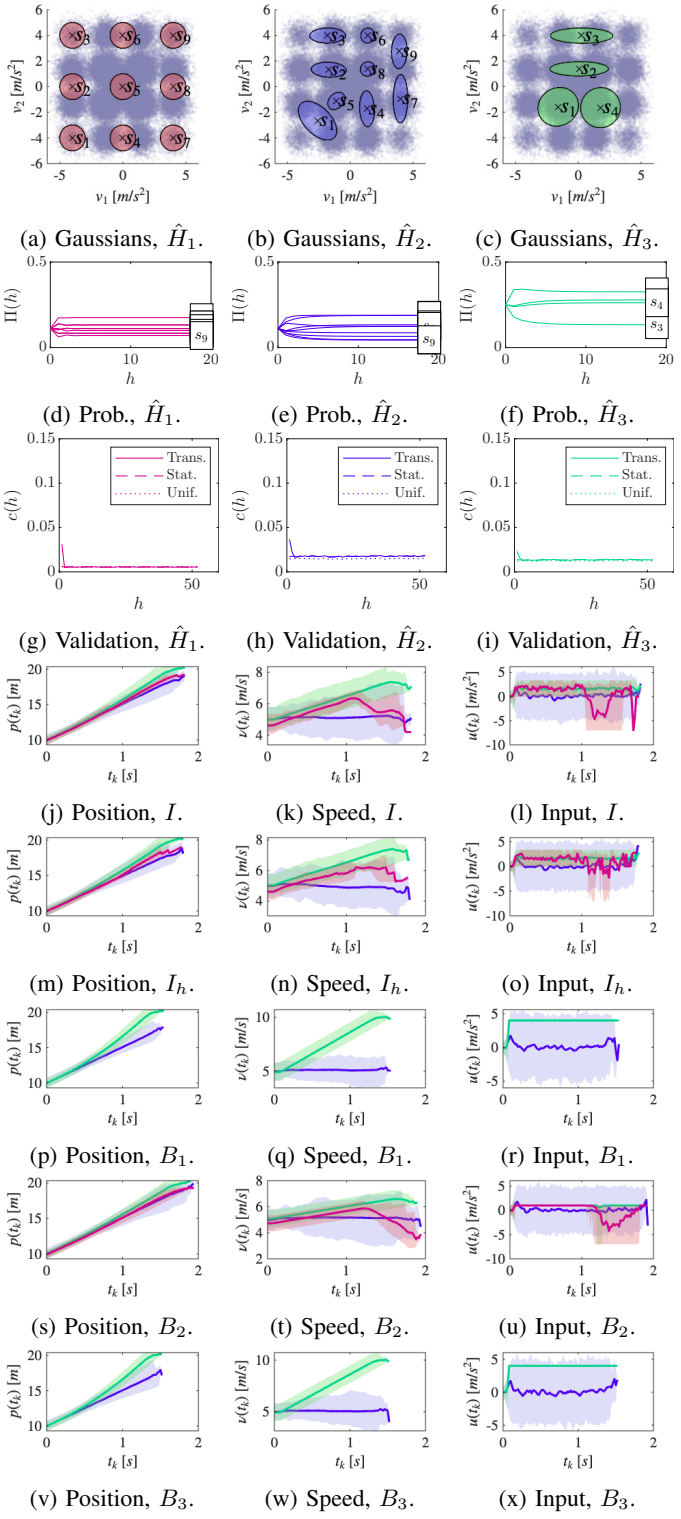


Fig. 8: Results, HDV_1 and HDV_2 uncertain.

brakes until it no longer predicts that a collision is possible. The control signals of I and I_h (shown in green shade in Figures 7l and 7o) fluctuate between states, due to different initial state probability estimates being from the noisy observed input of HDV_2. For all controllers, the AV gains speed and reaches the intersection before the HDVs, which is shown in blue in the speed graphs of Fig. 7.

B. HDV_1 and HDV_2 uncertain

All behavior state transition probabilities are now set to 0.5, making HDV_1 and HDV_2 maximally uncertain.

1) Learning: In this case, the validation scores in Figures 8h and 8i are low for both $\hat{H}_2$ and $\hat{H}_3$, and using DTMC based probabilities is no better than guessing uniformly between modes. The placement of $\hat{H}_2$'s and $\hat{H}_3$'s Gaussians is what makes the trained estimates slightly better predictors than $\hat{H}_1$.

2) Controller simulation: B_1 and B_3 are feasible in all cases. Figures 8r and 8x show that these controllers blindly maximize the acceleration, a successful albeit unsafe strategy for outrunning an uncertain driver. B_2 is feasible in 55% of simulations, and Fig. 8u suggests that B_2's modest acceleration and late emergency braking, shown in red, is the cause. I and I_h perform better, with 85% and 74% feasible simulations, respectively. This difference is likely due to I's use of integer collision avoidance constraints. As seen in Figures 8j and 8m, all feasible trajectories are from cases in which the AV passes before HDV_2, meaning that this option is always chosen over the less rewarding stopping option when both are available. However, these results suggest that the controller should only be used in situations where some scenarios are significantly more probable than other.

C. HDV_1 passive and HDV_2 aggressive

Behavior state probabilities are now set to $p_{i,a} = 0.1$ and $p_{i,p} = 0.9$ for HDV_1 and $p_{i,a} = 0.9$ and $p_{i,p} = 0.1$ for HDV_2, the opposite setting of the experiment in Section VII-A.

1) Learning: The data in Fig. 9a, 9b and 9c now shows that HDV_2 most often accelerates while HDV_1 brakes, as the roles of the HDVs are just reversed compared to the experiment in Section VII-A. Accordingly, Fig. 9 shows state probability developments and validation scores that are very close to those of the first experiment in Fig. 7.

2) Controller simulation: Now, the baselines perform poorly. B_1 and B_3, feasible in 56% and 65% of simulations, underestimate the velocity range of HDV_2 but succeed in passing it in some cases. B_2 is feasible in only 16% of simulations and likely overestimates the range and becomes infeasible due to HDV_2 behavior fluctuations. In contrast, the proposed controller I and its heuristic counterpart I_h produce feasible solutions in all cases. Predicting that the desired passing option will likely not be feasible, the control action is restricted to brake as HDV_1 does in the data. In Figures 9j and 9m, we observe that the controllers let HDV_2 pass before the AV.

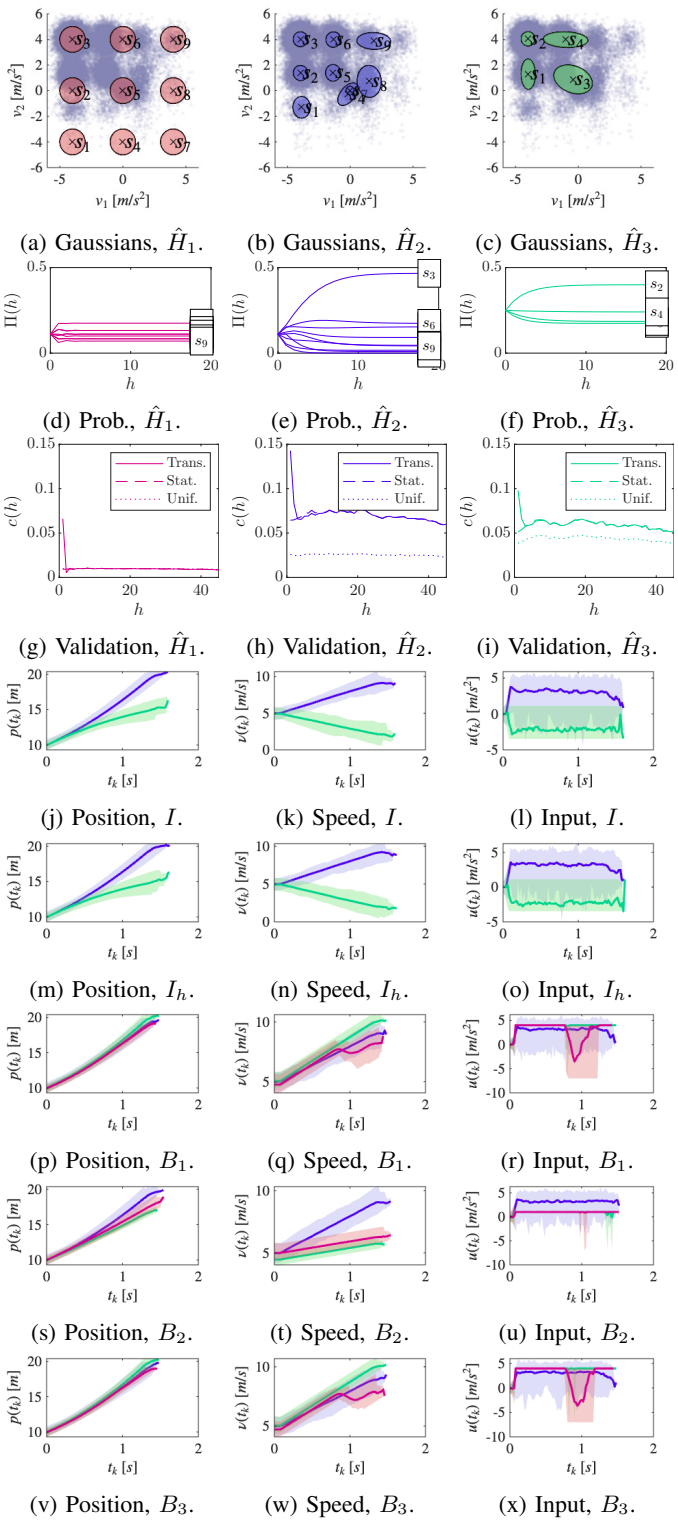


Fig. 9: Results, HDV_1 passive and HDV_2 aggressive.

VIII. Conclusion

Motion planning for AVs becomes increasingly difficult when control inputs are correlated with the input decisions of surrounding stochastic and interactive HDVs. To address this, we proposed an interaction-aware MPC that accounts for this correlation. We estimated HDV agent decision formation in a network as an HMM from input observations using a Baum-Welch-based algorithm, and predicted parameter sequences for a multivariate Gaussian that approximates the distribution of joint agent inputs. The interaction-aware MPC determines the control input by constrained optimization over all future scenarios resulting from applying the input ranges specified by the predicted Gaussian parameter sequences to the agents in the network. By also formulating input constraints from these parameters, the interactive behavior specified by the HMM is followed by the controlled agent.

In a traffic intersection case study, the interaction-aware MPC mirrored the behavior seen in the data used to estimate the underlying HMM. In scenarios where the distribution of

driver decisions was concentrated, the proposed controller outperformed the baselines. Although performance decreased in highly uncertain situations in which different decisions were equally probable, the proposed controllers I and I_h produced feasible solutions in 93% of simulations, while baseline MPCs B_1, B_2 and B_3 were feasible in 77% of simulations when comparing over all three experiments. Future work includes online HMM estimation, which can be used in an adaptive interaction-aware MPC strategy.

REFERENCES

[1] E. Yurtsever, J. Lambert, A. Carballo, and K. Takeda, "A survey of autonomous driving: Common practices and emerging technologies," *IEEE Access*, vol. 8, pp. 58 443–58 469, 2020.

[2] A. Rasouli and J. K. Tsotsos, "Autonomous vehicles that interact with pedestrians: A survey of theory and practice," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 3, pp. 900–918, 2020.

[3] L. Crosato, K. Tian, H. P. H. Shum, E. S. L. Ho, Y. Wang, and C. Wei, "Social interaction-aware dynamical models and decision-making for autonomous vehicles," *Advanced Intelligent Systems*, vol. 6, no. 3, p. 2300575, 2024. [Online]. Available: https://advanced.onlinelibrary.wiley.com/doi/abs/10.1002/aisy.202300575

[4] S. Kolekar, S. Gite, B. Pradhan, and K. Kotecha, "Behavior prediction of traffic actors for intelligent vehicle using artificial intelligence techniques: A review," *IEEE Access*, vol. 9, pp. 135 034–135 058, 2021.

[5] B. Brito, A. Agarwal, and J. Alonso-Mora, "Learning interaction-aware guidance for trajectory optimization in dense traffic scenarios," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 10, pp. 18 808–18 821, 2022.

[6] F. Camara, N. Bellotto, S. Cosar, D. Nathanael, M. Althoff, J. Wu, J. Ruenz, A. Dietrich, and C. W. Fox, "Pedestrian models for autonomous driving part i: Low-level models, from sensing to tracking," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 10, pp. 6131–6151, 2021.

[7] F. Camara, N. Bellotto, S. Cosar, F. Weber, D. Nathanael, M. Althoff, J. Wu, J. Ruenz, A. Dietrich, G. Markkula, A. Schieben, F. Tango, N. Merat, and C. Fox, "Pedestrian models for autonomous driving part ii: High-level models of human behavior," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 9, pp. 5453–5472, 2021.

[8] C.-J. Heiker and P. Falcone, "Decision modeling in markovian multi-agent systems," in *61st CDC*, 2022, pp. 7235–7240.

[9] P. Bolzern, P. Colaneri, and G. De Nicolao, "Opinion dynamics in social networks with heterogeneous markovian agents," in *IEEE CDC*, Miami, USA, Dec. 2018, pp. 6180–6185.

[10] P. Bolzern, P. Colaneri, and G. De Nicolao, "Opinion influence and evolution in social networks: A markovian agents model," *Automatica*, vol. 100, pp. 219–230, 2019.

[11] P. Bolzern, P. Colaneri, and G. De Nicolao, "Opinion dynamics in social networks: The effect of centralized interaction tuning on emerging behaviors," *IEEE Transactions on Computational Social Systems*, vol. 7, no. 2, pp. 362–372, 2020.

[12] P. Bolzern, P. Colaneri, and G. De Nicolao, "Effect of social influence on a two party election: A markovian multi-agent model," *IEEE Transactions on Control of Network Systems*, pp. 1–1, 2021.

[13] S. B. Amsalu, A. Homaifar, A. Karimoddini, and A. Kurt, "Driver intention estimation via discrete hidden markov model," in *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2017, pp. 2712–2717.

[14] S. Liu, K. Zheng, L. Zhao, and P. Fan, "A driving intention prediction method based on hidden markov model for autonomous driving," *Computer Communications*, vol. 157, pp. 143–149, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0140366419316639

[15] Y. Zhang, Y. Zou, Selpi, Y. Zhang, and L. Wu, "Spatiotemporal interaction pattern recognition and risk evolution analysis during lane changes," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 6, pp. 6663–6673, 2023.

[16] S. Lefèvre, Y. Gao, D. Vasquez, H. E. Tseng, R. Bajcsy, and F. Borrelli, "Lane Keeping Assistance with Learning-Based Driver Model and Model Predictive Control," in *12th International Symposium on Advanced Vehicle Control*, Tokyo, Japan, 2014. [Online]. Available: https://inria.hal.science/hal-01104458

[17] L. E. Baum *et al.*, "An inequality and associated maximization technique in statistical estimation for probabilistic functions of markov processes," *Inequalities*, vol. 3, no. 1, pp. 1–8, 1972.

[18] L. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[19] I. Batkovic, U. Rosolia, M. Zanon, and P. Falcone, "A robust scenario mpc approach for uncertain multi-modal obstacles," *IEEE Control Systems Letters*, vol. 5, no. 3, pp. 947–952, 2021.

[20] I. Batkovic, M. Ali, P. Falcone, and M. Zanon, "Safe trajectory tracking in uncertain environments," *IEEE Transactions on Automatic Control*, vol. 68, no. 7, pp. 4204–4217, 2023.

[21] V. Lefkopoulos and M. Kamgarpour, "Trajectory planning under environmental uncertainty with finite-sample safety guarantees," *Automatica*, vol. 131, p. 109754, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0005109821002740

[22] R. Wang, M. Schuurmans, and P. Patrinos, "Interaction-aware model predictive control for autonomous driving," in *2023 European Control Conference (ECC)*. IEEE, 2023, pp. 1–6.

[23] S. H. Nair, V. Govindarajan, T. Lin, C. Meissen, H. E. Tseng, and F. Borrelli, "Stochastic mpc with multi-modal predictions for traffic intersections," in *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*, 2022, pp. 635–640.

[24] K. Ren, H. Ahn, and M. Kamgarpour, "Chance-constrained trajectory planning with multimodal environmental uncertainty," *IEEE Control Systems Letters*, vol. 7, pp. 13–18, 2022.

[25] K. Ren, C. Chen, H. Sung, H. Ahn, I. M. Mitchell, and M. Kamgarpour, "Recursively feasible chance-constrained model predictive control under gaussian mixture model uncertainty," *IEEE Transactions on Control Systems Technology*, vol. 33, no. 4, pp. 1193–1206, 2024.

[26] G. Cesari, G. Schildbach, A. Carvalho, and F. Borrelli, "Scenario model predictive control for lane change assistance and autonomous driving on highways," *IEEE Intelligent Transportation Systems Magazine*, vol. 9, no. 3, pp. 23–35, 2017.

[27] G. Calafiore and M. Campi, "The scenario approach to robust control design," *IEEE Transactions on Automatic Control*, vol. 51, no. 5, pp. 742–753, 2006.

[28] G. Schildbach, L. Fagiano, C. Frei, and M. Morari, "The scenario approach for stochastic model predictive control with bounds on closed-loop constraint violations," *Automatica*, vol. 50, no. 12, pp. 3009–3018, 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0005109814004166

[29] H. Ahn, C. Chen, I. M. Mitchell, and M. Kamgarpour, "Safe motion planning against multimodal distributions based on a scenario approach," *IEEE Control Systems Letters*, vol. 6, pp. 1142–1147, 2022.

[30] C. G. Cassandras and S. Lafortune, "Markov chains," in *Introduction to discrete event systems*, 2nd ed. Springer, 2008, ch. 7, pp. 368–428.

[31] C.-J. Heiker, E. Gaetan, L. Giarré, and P. Falcone, "Repulsive markovian models for opinion dynamics," *Systems and Control Letters*, vol. 185, p. 105720, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167691124000082

[32] C.-J. Heiker and P. Falcone, "Trajectory planning among interactive markovian obstacles using scenario model predictive control*," in *2025 American Control Conference (ACC)*, 2025, pp. 8–13.

[33] R. Durbin, S. R. Eddy, A. Krogh, and G. Mitchison, *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press, 1998.

[34] Y. Chen, U. Rosolia, W. Ubellacker, N. Csomay-Shanklin, and A. D. Ames, "Interactive multi-modal motion planning with branch model predictive control," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 5365–5372, 2022.

[35] A. Bemporad and M. Morari, "Control of systems integrating logic, dynamics, and constraints," *Automatica*, vol. 35, no. 3, pp. 407–427, 1999. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0005109898001782

[36] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi – A software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, no. 1, pp. 1–36, 2019.

[37] P. Bonami, L. T. Biegler, A. R. Conn, G. Cornuéjols, I. E. Grossmann, C. D. Laird, J. Lee, A. Lodi, F. Margot, N. Sawaya *et al.*, "An algorithmic framework for convex mixed integer nonlinear programs," *Discrete optimization*, vol. 5, no. 2, pp. 186–204, 2008.