

GAMMA: A General Agent Motion Prediction Model for Autonomous Driving

Yuanfu Luo

Panpan Cai

David Hsu

Wee Sun Lee

Abstract—Autonomous driving in mixed traffic requires reliably predicting the motion of nearby *traffic agents*, such as pedestrians, bicycles, cars, buses, etc.. This prediction task is extremely challenging, because of the diverse geometry and dynamics of traffic agents, multi-way interactions among them, and complex road conditions. This paper presents GAMMA, a general agent motion prediction model for autonomous driving. GAMMA predicts the motion of heterogeneous traffic agents with different geometric, kinematics, and behavioral constraints. It formalizes motion prediction as constrained geometric optimization in the velocity space and integrates both *physical* and *behavioral* constraints into a unified framework. Our results show that GAMMA outperforms state-of-the-art motion prediction methods substantially on real-world datasets.

I. INTRODUCTION

Autonomous vehicles navigating in dynamic and interactive environments like Fig. 1 need to foresee the future motion of nearby *traffic agents* (pedestrians, bicycles, cars, buses, etc.) to ensure safety and efficiency. However, it is extremely challenging to predict the motions of real-world traffic agents, due to the diverse dynamics and geometry of traffic agents, complex road conditions, and intensive interactions among the agents. We observe that the motion of traffic agents are governed by various physical and behavioral constraints. Traffic agents try to reach their destinations under their kinematic constraints, and in the meantime, they share the responsibility for collision avoidance with each other, while having limited attention capabilities.

In this paper, we identify five key factors that govern traffic agents' motions: *kinematics*, *geometry* (collision avoidance), *intention* (destination), *responsibility*, and *attention*. We develop GAMMA, a General Agent Motion prediction Model for Autonomous driving, that integrates these factors in a unified framework. GAMMA formalizes motion prediction as geometry optimization in the velocity space, which is conditioned on physical constraints (kinematics and geometry) and behavioral constraints (intentions, attentions, and responsibilities). The optimization objective is specified by the intention of a traffic agent, while the constraints are imposed by kinematics, geometry, responsibilities, and attentions.

GAMMA significantly extends the existing constrained geometry optimization framework [1] in the following aspects. First, GAMMA explicitly models heterogeneous kinematics. It enforces the predictions to be kinematically feasible by only considering velocities that can be tracked within a maximum tracking error with the agent's kinematics. Second, GAMMA uses polygon representation for agents' geometry,

The authors are with School of Computing, National University of Singapore. {yuanfu, caipp, dyhsu, leews}@comp.nus.edu.sg



Fig. 1. Screenshots of our heterogeneous datasets: UTOWN and CROSS.

which is tighter than disc-shaped ones in [1] and much more representative for most real-world traffic agents. Third, GAMMA incorporates various behavioral constraints to the model. Since behavioral constraints are hidden and differ from individuals, GAMMA applies Bayesian inference to infer a distribution over possible behavioral constraints for each agent. It then generates trajectories using the behavioral constraints of maximum likelihood.

We collected two datasets containing heterogeneous traffic agents (Fig. 1). We compare GAMMA with state-of-the-art motion prediction models on our new datasets and two standard benchmark datasets. Results show that GAMMA significantly outperforms these approaches in predicting real-world trajectories. Our ablation study further demonstrates the importance of the identified key factors in trajectory predictions. Our source code and datasets will be released upon paper acceptance.

II. RELATED WORK

A. Traditional Approaches For Motion Prediction

We categorize the traditional approaches for motion prediction into four groups: social force based, geometry optimization based, and maneuver recognition based, and rule based approaches. Social force models [2], [3], [4], [5], [6] assume that traffic agents are driven by virtual forces generated from the internal motivations such as reaching the goal, and the external constraints such as avoiding obstacles. Geometry optimization based approaches compute collision-free motions for multiple traffic agents via optimization in the feasible geometric space. The representative work includes Velocity Obstacle (VO) based [7] and the Reciprocal Velocity Obstacle (RVO) based algorithms [8], [1], [9]. Maneuver recognition based approaches [10], [11], [12] first classify the traffic agents' motions into semantically interpretable maneuver using approaches such as HMMs and SVMs, and then predict the motions conditioned on the maneuver with traditional approaches, e.g., polynomial curve fitting [13], minimization on carefully designed cost functions, etc.. Rule based approaches predict motions assuming traffic agents move by simple rules such as lane following. Many driving simulators [14], [15], [16] use rule-based motion models.

These approaches, however, do not explicitly model the five factors we proposed and hence are not able to handle heterogeneous traffic agents with different kinematics, geometry representations, or behavioral constraints.

Some previous work has considered some of the five factors. NH-ORCA [17] and B-ORCA [18] predict motions considering non-holonomic constraints. AVO [19] takes into account the acceleration constraints to ensure continuous velocity. Those approaches, however, do not generalize to general kinematics, hence cannot handle the interactions between traffic agents with different kinematic constraints. GVO [20] uses a general algebraic representation to incorporate different kinematics. However, it models poorly the interactions among traffic agents since it assumes all other agents are non-reactive. Some previous work also tried different geometry representations for agents in collision avoidance, such as ellipse [21] and capsule [22]. Ma et al. [23] present a novel CTMAT representation based on medial axis transformation to compute tight bounding shapes for different agents. Our work uses polygon representation because it is representative for most traffic agents and easy for collision checking. Some previous work has modeled behavioral constraints. PORCA [24] infers intentions with Bayesian inference, but it assumes responsibility changes deterministically and all the agents are fully attentive. The work in [25] learns a linear mapping from selected features to attention values. The mapping, however, is learned for cars and might not be suitable for traffic agents of other types.

B. Deep Learning Approaches For Motion Prediction

Considerable research has been done for motion prediction with deep learning approaches. S-LSTM [26] models each agent's trajectory with one LSTM, and models their interactions using a "social pooling layer". S-GAN [27] improves on S-LSTM by introducing GAN to it. SoPhie [28] leverages a social attention module and a physical attention module to predict trajectories. SRLSTM [29] integrates a state refinement module into LSTM for joint trajectory predictions. These four works achieve high accuracy for motion prediction. However, they are designed specifically for pedestrians and are not suitable for other traffic agents such as cars. Another group of work focuses on vehicle trajectory prediction [30], [31], [32], [33]. They are all based on recurrent neural network. One limitation of this group of work is again the motion prediction for only one type of traffic agents. They may not work well for heterogeneous traffic agents with different kinematics and geometry. TraPHic [34] and TrafficPredict [35] consider heterogeneous traffic agents with different kinematics and geometry. TraPHic achieves this by embedding dynamics- and shape-relevant features in the input state space, while TrafficPredict achieves this by using "category layers" to learn the similarities of traffic agents of the same type. All these deep learning approaches do not model behavioral constraints explicitly. Instead, they directly model the interactions in a deep learning fashion. This often results in overfitting to the training scenes, which we will demonstrate in the experiment section. Another com-

mon issue of the deep learning approaches is the requirement of large training data; but training data is often hard to obtain.

III. GAMMA

The motion of traffic agents are determined by *physical constraints*, such as kinematics and geometry, and *behavioral constraints* such as intention, attention, and responsibility. Our motion prediction model, GAMMA, integrates both of them into a unified framework of constrained geometry optimization. It assumes each traffic agent optimizes its velocity based on its intention while constrained by kinematics, geometry, attention, and responsibility. In the following, we will first present our optimization framework, and then introduce in detail how we embed different factors to it.

A. Optimization Framework

We formulate motion prediction as a constrained geometry optimization problem. Given a group of traffic agents, we compute for each agent an instantaneous velocity close to its intended one, which is collision-free and feasible for the traffic agent's kinematics. We introduce the feasible space and objective function of the problem in the following.

1) *Feasible Velocity Set*: For a given traffic agent, GAMMA finds a set of velocities that is *kinematically trackable* and *geometrically feasible* for the traffic agent. A velocity v for a traffic agent A is defined as geometrically feasible with respect to another traffic agent B if it does not lead to collisions between them supposing that A moves at v for τ time. It is called kinematically trackable if A can track v with its low-level controller below a predefined maximum tracking error ε_{\max} for τ time. We denote the set of kinematically trackable velocities as K_A^τ , the set of geometrically feasible velocities for agent A with respect to agent B as $G_{A|B}^\tau$. Moreover, traffic agents only have limited attention capabilities. A real-world traffic agent only reacts to agents within its attentive range, denoted as $\text{Att}(A)$. Hence, GAMMA finds the optimal velocity for a traffic agent A from $G_A^\tau \cap K_A^\tau$, where $G_A^\tau = \bigcap_{B \in \text{Att}(A)} G_{A|B}^\tau$.

2) *Velocity Optimization*: To predict the motion of A , GAMMA computes a velocity that is closest to its *preferred velocity* v_A^{pref} derived from A 's intention. This optimization is conducted on the feasible velocity set $G_A^\tau \cap K_A^\tau$, i.e.,

$$v_A^{\text{new}} = \arg \min_{v \in G_A^\tau \cap K_A^\tau} \|v - v_A^{\text{pref}}\|. \quad (1)$$

The objective function here is quadratic. Moreover, both G_A^τ and K_A^τ are convex because of the way they are constructed (Sec. III-B). Therefore, we can efficiently solve the optimization problem using linear programming. The actual prediction of A 's motion is generated by tracking v_A^{new} using A 's low-level controller for one time step. This process repeats for t_{pred} steps for all traffic agents at time t to predict their future positions $\mathbf{P}_{t+1:t+t_{\text{pred}}}$.

B. Modeling Physical and Behavioral Constraints

GAMMA integrates both physical and behavioral constraints into the optimization framework. We will introduce the modeling of the following five factors in detail: kinematics, geometry, intention, attention, and responsibility.

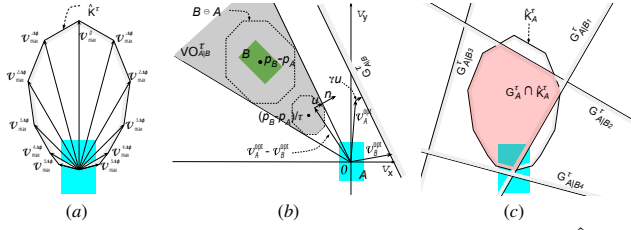


Fig. 2. (a) The estimated kinematically trackable velocity set, \hat{K}^τ , is estimated using the convex hull of $\{v_{\max}^\phi \mid \phi \in \Phi\}$. (b) $\text{VO}_{A|B}^\tau$ (gray) and $G_{A|B}^\tau$ (half plane). The velocity obstacle of agent A (blue) induced by agent B (green) for time τ , is a truncated cone with its apex at the origin (in velocity space) and its legs tangent to the polygon $B \ominus A$. $G_{A|B}^\tau$ is a half plane divided by the line perpendicular to the vector u through the point $v_A^{\text{opt}} + \gamma u$, where u is the vector from $v_A^{\text{opt}} - v_B^{\text{opt}}$ to the closest point on the boundary of $\text{VO}_{A|B}^\tau$. (c) An example of the feasible velocity space (red) of agent A . It is the intersection of \hat{K}_A^τ and the geometry constraints induced by four other agents: $G_{A|B_1}^\tau, G_{A|B_2}^\tau, G_{A|B_3}^\tau$, and $G_{A|B_4}^\tau$.

1) Kinematics Modeling: The motions of real-world traffic agents are constrained by their versatile types of kinematics. For example, most vehicles cannot move sidewise. To consider the influence of kinematics on agents' motions, we follow the ideas in [17] and [18] to introduce additional kinematic constraints to the feasible velocity space. GAMMA only chooses velocities that can be tracked by the agent's kinematics within a maximum tracking error ε_{\max} . The final prediction is generated by tracking the velocity output by GAMMA using a low-level controller. Note that both [17] and [18] are specifically designed for one single type of kinematics. The challenge here is to model traffic agents with *heterogeneous* kinematics, such as holonomic (pedestrians), car-like (cars, bicycles), trailer-like (trucks), etc..

For a given traffic agent A , we need to find the *kinematically trackable velocity set*, K_A^τ , s.t.,

$$K_A^\tau = \{v \mid \|vt - f_A(v, t)\| \leq \varepsilon_{\max}, \forall t \in [0, \tau]\}, \quad (2)$$

where $f_A(v, t)$ is the position of A at time t if it tracks v with its low-level controller. $f_A(v, t)$ varies for different types of kinematics and low-level controllers. It is often intractable to compute an analytic form of $f_A(v, t)$ for different kinematics and controllers. We propose to estimate K_A^τ numerically offline. We assume that the same type of traffic agents corresponds to the same kinematics and low-level controllers. Therefore, we only need to estimate K^τ 's for representative types of traffic agents: pedestrian, bicycle, motorbike, car, van, bus, gyro-scooter, trucks, etc..

For a specific type of traffic agents, we estimate its K^τ by first discretizing the set of holonomic velocities $\{v\}$ and running the controller offline to measure the maximum error ε_v for tracking v . Then K^τ is estimated as the convex hull of the discretized velocities with $\varepsilon_v \leq \varepsilon_{\max}$.

Concretely, we discretize v along its two dimensions (s_v, ϕ_v) , where s_v is the speed and ϕ_v is the deviation angle from traffic agent's heading direction, into two discrete sets $S = [0 : \Delta s : s_{\max}]$ and $\Phi = [0 : \Delta \phi : \phi_{\max}]$. This forms a discretized velocity set $V = \{v \mid s_v \in S, \phi_v \in \Phi\}$. For each $v \in V$, we run the controller offline to track it for a duration of τ and record its maximum tracking error ε_v . Then we determine the velocities on the boundary

of the trackable set as follows. For each deviation angle $\phi \in \Phi$, we collect a set V_ϕ of discretized velocities with deviation angle ϕ and has a tracking error less than ε_{\max} , i.e., $V_\phi = \{v \in V \mid \phi_v = \phi, \varepsilon_v \leq \varepsilon_{\max}\}$. After that, we pick a velocity v_{\max}^ϕ from V_ϕ that has the maximum speed, i.e., $v_{\max}^\phi = \arg \max_{v \in V_\phi} s_v$. Then v_{\max}^ϕ for all ϕ 's form the boundary of the trackable velocity set, and we approximate K^τ by the convex hull of these boundary velocities:

$$\hat{K}^\tau = \text{ConvexHull}(\{v_{\max}^\phi \mid \phi \in \Phi\}). \quad (3)$$

Geometrically, \hat{K}^τ is a polygon in velocity space (Fig. 2a). Although approximating K^τ brings a small approximation error, we claim that GAMMA is robust to this error.

2) Geometry Modeling: For a given agent A , we construct $G_{A|B}^\tau$, the geometrically feasible velocity set of A with respect to another agent B . We extend the definitions in [1] to handle agents with more general geometry. GAMMA first constructs a velocity obstacle for A with respect to B , then derives $G_{A|B}^\tau$ based on the velocity obstacle.

Consider two agents A and B at position \mathbf{p}_A and \mathbf{p}_B , respectively. The velocity obstacle $\text{VO}_{A|B}^\tau$ is defined as the set of all *relative* velocities of A with respect to B that will result in collisions before time τ . Formally,

$$\text{VO}_{A|B}^\tau = \{v \mid \exists t \in [0, \tau], t \cdot v \in (B \ominus A)\}, \quad (4)$$

where $B \ominus A$ represents the *Minkowski difference*, which essentially inflates the geometry of B by that of A , and treats A as a single point. Note that the definition of velocity obstacle in [1] assumes the geometry of traffic agents to be disc-shaped. This loose representation leads to overly conservative motions in crowded scenes, especially for agents with large aspect ratios such as cars and bicycles. Instead, GAMMA adopts a polygon representation, which fits tighter with most real-world traffic agents. Fig. 2b visualizes a velocity obstacle constructed with two polygon-shaped agents.

We construct $G_{A|B}^\tau$ with respect to the *optimization velocities* of the agents (often set to their current velocities) similar to [1]. Consider the case where A and B will collide with each other before time τ by taking their optimization velocities v_A^{opt} and v_B^{opt} . To avoid collisions with the least cooperative effort, GAMMA finds a relative velocity closest to $v_A^{\text{opt}} - v_B^{\text{opt}}$ from the boundary of $\text{VO}_{A|B}^\tau$. Let u be the vector from $v_A^{\text{opt}} - v_B^{\text{opt}}$ to this optimal relative velocity:

$$u = \left(\arg \min_{v \in \partial \text{VO}_{A|B}^\tau} \|v - (v_A^{\text{opt}} - v_B^{\text{opt}})\| \right) - (v_A^{\text{opt}} - v_B^{\text{opt}}). \quad (5)$$

Then u is the smallest change on the relative velocity to avoid the collision within τ time. GAMMA lets A take $\gamma \in [0, 1]$ of the responsibility for collision avoidance, i.e., adapt its velocity by $\gamma \cdot u$. Then $G_{A|B}^\tau$ is constructed as a half plane:

$$G_{A|B}^\tau = \{v \mid (v - (v_A^{\text{opt}} + \gamma \cdot u)) \cdot n \geq 0\}, \quad (6)$$

where n is the outward normal at point $(v_A^{\text{opt}} - v_B^{\text{opt}}) + u$. Fig. 2b visualizes $G_{A|B}^\tau$.

Since $G_{A|B}^\tau$ is a half plane in the velocity space, $G_{A|B}^\tau = \bigcap_{B \in \text{Att}(A)} G_{A|B}^\tau$ is convex. Moreover, since K_A^τ is convex, the feasible velocity space $G_A^\tau \cap K_A^\tau$ will also be convex. Fig. 2c visualizes an example feasible space induced by both

K_A^τ and G_A^τ considering four other traffic agents.

3) *Intention Modeling*: GAMMA computes a preferred velocity v^{pref} for each agent based on its intention, and uses v^{pref} as the optimization target (Eq. 1). The intention of a traffic agent is commonly represented as navigating to a goal location, where the set of possible goals are known a priori for each environment [1], [24]. This intention representation limits its application to complex real-world environments in the following aspects. First, we need to re-specify the goal locations for each new environment, which requires a lot of manual work. Second, the actual goals of agents may not be covered by the predefined set, resulting in inaccurate predictions. Third, since agents target at a fixed goal location and GAMMA only performs local optimization, an agent can be easily trapped by large obstacles blocking its goal.

We propose a new intention representation to address these issues. We define the intention of an agent as whether it wants to keep moving at current velocity or at current acceleration. This design is based on two observations: 1) constant velocity model often predicts well when an agent intends to go straight, and 2) constant acceleration model often predicts well when the agent intends to turn. Based on the observations, we apply Bayesian inference to infer each agent's actual intention using its history positions (Sec. III-B.6). Given an inferred intention, we generate a reference position for the agent by applying the current velocity/acceleration for t_{pred} time. This position defines the preferred velocity v^{pref} of the agent, by assigning a direction pointing to the position and a speed equal to its current speed.

4) *Attention Modeling*: Real-world traffic agents only have limited attention capabilities. An agent does not pay uniform attention to all nearby agents. For instance, it reacts more to the agents in front because they are critical to collision avoidance. Different agents usually have different *attentive radii*, i.e., the radius within which other agents will be noticed. For example, aggressive agents often have smaller attentive radii, while conservative agents have larger ones.

According to above discussions, GAMMA adopts a half-circle attention mechanism. It uses two half circles to model the attention: one in front of the vehicle with radius r_{front} , and one at the back with radius r_{rear} . We set $r_{\text{rear}} \leq r_{\text{front}}$, giving more attention to agents in the front. The actual values of r_{front} and r_{rear} are also determined using Bayesian inference (Sec. III-B.6). Particularly, we define a small set of typical values for r_{front} and r_{rear} , and then infer for each traffic agent the most likely values using its history.

Let $\text{Att}(A)$ denote the agents inside the inferred half circles of A . GAMMA only models the interactions between A and agents in $\text{Att}(A)$. Namely, it only considers the constraints induced by a traffic agent B if B is in $\text{Att}(A)$.

5) *Responsibility Modeling*: The responsibility that an agent is willing to take for reciprocal collision avoidance is correlated to its type and its distance to the interacting agents. A vehicle interacting with a pedestrian often takes more responsibility for collision avoidance when they are far apart. But when they approach each other, the pedestrian will take more responsibility because it is more flexible in

side-wise movements. We assume that the responsibility of an agent changes linearly with its distance to other agents: $\gamma = C_1 \cdot d + C_2$, where γ is the responsibility and d is the distance. The coefficient C_2 determines the initial responsibility and C_1 is the responsibility changing rate.

This model is similar to that in [24]. But now the coefficients, C_1 and C_2 , differ for each agent and need to be inferred. We predefined a set of values of C_1 and C_2 for each type of traffic agents based on our thumb of rule that, the more physical constraints a traffic agent has, the larger its initial responsibility (C_2) and the smaller its responsibility changing rate (C_1) are. Then, we apply Bayesian inference (Sec. III-B.6) to infer their values based on the history positions. To use the inferred responsibility, we first normalize the value of the two agents so that they sum up to one. Then, the responsibility is injected into Eq. 6 to construct the pair-wise collision-avoiding half planes. Note that GAMMA can model obstacles as static agents by assigning them a responsibility of 0 and a preferred velocity of 0.

6) *Bayesian Inference for Behaviors*: Behavioral constraints like intention, attention and responsibility are not observable. We can only infer a distribution over their values from agents' history positions. We apply Bayesian inference to achieve this. For each history time step, we execute GAMMA with a fixed set of behavioral constraints to simulate the stochastic dynamics of an agent and compute the likelihood of its actual movement. This is repeated for all combinations of behavioral constraints to get all likelihood. Then, we apply the Bayes' rule to incorporate the likelihood, update the prior, and obtain a posterior distribution.

Formally, given a particular set of behavioral constraints:

$$\mathcal{B} = (\text{intent}, r_{\text{front}}, r_{\text{rear}}, C_1, C_2), \quad (7)$$

we provide a stochastic estimation for the next position of the agent, by executing GAMMA to get a mean position $\mathbf{p}^{\text{GAMMA}}$ and adding a Gaussian noise to it:

$$\mathbf{p}_t = \mathbf{p}_t^{\text{GAMMA}}(\mathcal{B}, \mathbf{P}_{t-t_{\text{hist}}:t-1}, \mathbf{T}, \mathbf{O}) + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma^2), \quad (8)$$

where $\mathbf{P}_{t-t_{\text{hist}}:t-1}$ are history positions, \mathbf{T} is the set of agent types, and \mathbf{O} is the set of static obstacles in the environment. The Gaussian noise has a predefined variance σ^2 . We assume that $\mathbf{P}_{t-t_{\text{hist}}:t-1}$, \mathbf{T} , and \mathbf{O} are fully observable because they are relatively accurate and their noises do not pose significant effect on motion prediction compared to behavioral constraints. We also assume that a traffic agent does not change its behavior during the prediction horizon.

For an observed agent position \mathbf{p}_t at a history time step t , the observation likelihood is computed as:

$$p(\mathbf{p}_t | \mathcal{B}, \mathbf{P}_{t-t_{\text{hist}}:t-1}, \mathbf{T}, \mathbf{O}) = f(\|\mathbf{p}_t - \mathbf{p}_t^{\text{GAMMA}}\| | 0, \sigma^2),$$

where f is the probability density function of the normal distribution. Next, we update the probability distribution over \mathcal{B} at time step t using the Bayes' rule:

$$p_t(\mathcal{B}) = \eta \cdot p(\mathbf{p}_t | \mathcal{B}, \mathbf{P}_{t-t_{\text{hist}}:t-1}, \mathbf{T}, \mathbf{O}) \cdot p_{t-1}(\mathcal{B}), \quad (9)$$

where $p_{t-1}(\mathcal{B})$ is the prior distribution, $p_t(\mathcal{B})$ is the posterior at time step t , and η is a normalization constant. After parsing through all history positions of an agent, we obtain

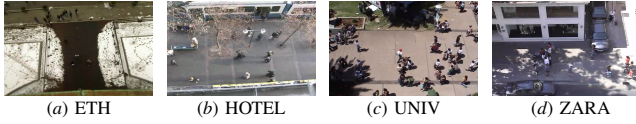


Fig. 3. Screenshots of homogeneous datasets. (a) and (b) are ETH dataset. (c) and (d) are UCY dataset.

an informed distribution over its behavioral constraints.

IV. EXPERIMENTS AND RESULTS

In this section, we evaluate our method GAMMA on four datasets: ETH[36], UCY[37], UTWON, and CROSS. We first evaluate GAMMA on *homogeneous* datasets with only pedestrians (ETH and UCY) in Sec. IV-A, then on *heterogeneous* datasets with various types of agents (UTOWN and CROSS) in Sec. IV-B. We also conduct an ablation study in Sec. IV-C, and a speed comparison in Sec. IV-D.

Datasets. ETH and UCY are standard benchmark datasets. ETH consists of two scenes: a plaza outside the ETH building (ETH) and a street outside a hotel (HOTEL); UCY also consists of two scenes: a university plaza (UNIV), and a street outside Zara (ZARA1 and ZARA2). However, the two datasets contain only pedestrians. To test our predictor on heterogeneous traffic agents, we further collected two datasets: UTOWN and CROSS. UTOWN presents a campus plaza where many pedestrians interacting with a vehicle scooter. CROSS presents an *unsignalized* cross scene with various types of traffic agents, including cars, vans, buses, bicycles, motorcycles, electric-tricycles, and pedestrians. See Fig. 3 for the screenshots of the dataset scenes.

Evaluation Metrics. Following prior work [32], [27], we report two error metrics:

- Average Displacement Error (**ADE**). The average Euclidean distance between the predicted position and the ground-truth position over all the time frames for a prediction duration $t_{\text{pred}} = 4.8s$.
- Final Displacement Error (**FDE**). The Euclidean distance between the predicted position and the ground-truth position at the final time frame ($t_{\text{pred}} = 4.8s$).

Baselines. We compare GAMMA with following baselines:

- LR: A linear regressor minimizing least square error over history trajectories.
- S-LSTM [26]: Social LSTM which models each trajectory with one LSTM, and models their interactions using a social pooling layer.
- SRLSTM [29]: State Refinement LSTM which embeds a state refinement module into LSTM to address the problem of joint trajectory predictions.
- TrafficPredict [35]: LSTM-based motion predictor for heterogeneous traffic agents. Note that TrafficPredict in [35] reports isometrically normalized results. We scale them back for consistent comparison.
- S-Force [6]: Social-force approach using an energy function to generate next actions for agents considering their preferred velocities, collisions, etc..
- PORCA [24]: Geometric optimization based approach similar to GAMMA, but does not consider the heterogeneous kinematics and geometry.

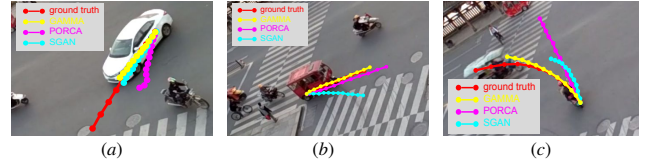


Fig. 4. The trajectory predictions of different algorithms compared with the ground truth (red). (a) PORCA predicts an infeasible trajectory (pink). (b) SGAN predicts a trajectory pointing to a wrong direction (blue). (c) GAMMA prediction (yellow) is closest to the ground truth.

- SGAN [27]: Social GAN which introduces generative adversarial network into S-LSTM.
- SoPhie [38]: GAN based approach that leverages both social and physical attention mechanisms to consider interactions and physical constraints of the agents.
- MATF [39]: Multi-Agent Tensor Fusion network that fuses information from a static scene context with multiple dynamic agent states.
- S-Ways [40]: Social-Ways which uses Info-GAN[41] for trajectory predictions.

A. Evaluation on Homogeneous Datasets

We compare GAMMA with state-of-the-art approaches on homogeneous benchmark datasets, ETH and UCY, which only contain pedestrians. SGAN, MATF, SoPhie, and S-Ways are stochastic models. They all generate 20 samples and choose the one closest to ground truth for evaluation. For a fair comparison, we modified GAMMA and get its stochastic variant, GAMMA-s. GAMMA-s generates 20 samples by sampling Gaussian noise on the inferred goal positions and chooses the closest sample. Choosing the closest sample, however, is meaningless for autonomous driving since it requires the hindsight knowledge of the ground truth. One way to address this issue is to compute an average sample and use the average sample as the prediction. Following this idea, we modified SGAN and get its variant SGAN-ave.

The ADE and FDE of GAMMA and the baselines are shown in Table I. GAMMA outperforms all the baselines, especially, the state-of-the-art deterministic models SRLSTM and PORCA, and stochastic models MATF and S-Ways in almost all datasets. Although the performance gain of GAMMA over PORCA is not very significant here, GAMMA outperforms PORCA very significantly on datasets with heterogeneous traffic agents (Sec. IV-B).

One interesting finding is that, the simple model LR significantly outperforms many deep learning approaches including LSTM, S-LSTM, SGAN and SoPhie, in the HOTEL scene, which is less crowded and mostly contains straight-line trajectories. It demonstrates that those complex models might overfit to the more complex scenes. In contrast, our approach performs well in both simple and complex scenes.

B. Evaluation on Heterogeneous Datasets

We evaluate GAMMA on the heterogeneous datasets, UTOWN and CROSS, which contain various types of traffic agents such as pedestrians, cars, and bicycles. Our baselines include SRLSTM, the best-performing *deep learning* baseline for homogeneous datasets, and PORCA, the best-performing *traditional* baseline for homogeneous datasets.

TABLE I

PERFORMANCE COMPARISON ACROSS HOMOGENEOUS BENCHMARK DATASETS, ETH (ETH AND HOTEL) AND UCY (UNIV, ZARA1, AND ZARA2). THE ADE AND FDE VALUES ARE SEPARATED BY SLASH. THEIR AVERAGE VALUES (AVG) ARE ALSO REPORTED.

Dataset	Deterministic Models							Stochastic Models					
	LR	S-LSTM	SRLSTM	TrafficPredict	S-Force	PORCA	GAMMA	SGAN	SGAN-ave	SoPhie	MATF	S-Ways	GAMMA-s
ETH	1.33/2.94	1.09/2.35	0.63/1.25	5.46/9.73	0.67/1.52	0.52/1.09	0.51/1.08	0.81/1.52	0.96/1.87	0.70/1.43	1.01/1.75	0.39/0.64	0.43/0.91
HOTEL	0.39/0.72	0.79/1.76	0.37/0.74	2.55/3.57	0.52/1.03	0.29/0.60	0.28/0.59	0.72/1.61	0.61/1.31	0.76/1.67	0.43/0.80	0.39/0.66	0.23/0.48
UNIV	0.82/1.59	0.67/1.40	0.41/0.90	4.32/8.00	0.74/1.12	0.47/1.09	0.44/1.06	0.60/1.26	0.57/1.22	0.54/1.24	0.44/0.91	0.55/1.31	0.43/1.03
ZARA1	0.62/1.21	0.47/1.00	0.32/0.70	3.76/7.20	0.40/0.60	0.37/0.87	0.36/0.86	0.34/0.69	0.45/0.98	0.30/0.63	0.26/0.45	0.44/0.64	0.32/0.75
ZARA2	0.77/1.48	0.56/1.17	0.51/1.10	3.31/6.37	0.40/0.68	0.30/0.70	0.28/0.68	0.42/0.84	0.39/0.86	0.38/0.78	0.26/0.57	0.51/0.92	0.24/0.59
AVG	0.79/1.59	0.72/1.54	0.45/0.94	3.88/6.97	0.55/0.99	0.39/0.87	0.37/0.85	0.58/1.18	0.60/1.25	0.54/1.15	0.48/0.90	0.46/0.83	0.33/0.75

TABLE II

PERFORMANCE COMPARISON ACROSS HETEROGENEOUS DATASETS CROSS AND UTOWN. ADE/FDE, AND THEIR MEAN (AVG) ARE REPORTED.

Dataset	TrafficPredict	SRLSTM	SGAN-ave	SGAN	PORCA	GAMMA
CROSS	6.49/11.56	1.36/3.17	1.16/2.66	0.93/2.18	0.89/2.10	0.64/1.77
UTOWN	2.82/4.24	0.41/0.96	0.78/1.88	0.59/1.43	0.34/0.86	0.32/0.84
AVG	4.66/7.90	0.97/2.27	0.76/1.81	0.76/1.81	0.62/1.48	0.48/1.31

TABLE III

ABLATION STUDY FOR GAMMA ON CROSS. ADE/FDE ARE REPORTED.

w/o kinematic constraints	w/o polygon representation	w/o intention inference	w/o attention inference	w/o responsibility inference	GAMMA
0.68/1.84	0.84/2.07	0.96/2.29	0.65/1.80	0.65/1.77	0.64/1.77

We also compare GAMMA with TrafficPredict, which is designed specifically for heterogeneous traffic agents. SGAN and SGAN-ave are also added for comparison.

We show the ADE and FDE for all algorithms in Table II. Clearly, GAMMA outperforms all the baselines, especially for the challenging CROSS dataset, where various types of traffic agents move in a highly interactive manner. See the supplementary material or <https://youtu.be/aNWZdCdCLOM> for a video of prediction on CROSS dataset.

We further select three example scenes in CROSS and visualize the predictions in Fig. 4. Fig. 4a explains why PORCA does not predict well: it does not consider kinematic constraints and hence sometimes produces infeasible trajectories. Fig. 4b explains why SGAN does not predict well: it does not explicitly model intentions and hence sometimes produces trajectories pointing to wrong directions. Fig. 4c shows GAMMA successfully predicts a curving trajectory to the goal because it takes into account both the kinematics and the intentions while PORCA and SGAN fail to do so.

C. Ablation Study

To study the importance of the five components of the motion model, we conduct an ablation study. We test GAMMA in the absence of each component while holding others constant. For GAMMA w/o kinematic constraints, we set all agents to be holonomic. For GAMMA w/o polygon representation, we set all agents to be disc-shaped. For GAMMA w/o intention inference, we set the intention of all the agents to keeping moving at current velocity. For GAMMA w/o attention inference, we set the attention of all the agents to be 1. For GAMMA w/o responsibility inference, we set the responsibility of all the agents to be 0.5. We select CROSS as the testing dataset. The results are presented

TABLE IV

SPEED (IN SECOND) COMPARISON WITH SGAN AND SRLSTM OF DIFFERENT BATCH SIZE (bs). THE AVERAGE TIME EACH APPROACH TAKES TO PREDICT ONE TRAJECTORY IS REPORTED. WE ACHIEVE 8.04X SPEEDUP AS COMPARED TO SGAN ($bs = 1$). ALL APPROACHES ARE BENCHMARKED ON A WORKSTATION WITH INTEL(R) CORE(TM) I7-7700K CPU AND NVIDIA GTX 1080 GPU.

	SGAN ($bs = 1$)	SGAN ($bs = 64$)	SRLSTM ($bs = 1$)	SRLSTM ($bs = 4$)	GAMMA
run time	3.15e-3	1.80e-3	2.32e-3	2.19e-3	3.92e-4
speed-up	1x	1.75x	1.36x	1.44x	8.04x

in Table III. As we can see, each of the five components contributes to the performance gain of GAMMA. Kinematic constraints, polygon representation, and intention inference play a more important role in improving the model.

D. Speed Comparison

A motion prediction model needs to run sufficiently fast so that it can be used in autonomous driving which requires the robot vehicle to predict the myriad future motions. We compare the speed of our approach with those of two baselines, SGAN and SRLSTM. In reality, we have to predict the motions in chronological order since the observations for predictions come in chronological order. Therefore, for a fair and reasonable comparison, we set the batch size of SGAN and SRLSTM to 1 to avoid predicting trajectories before the observations actually come. We compare their average running time for predicting one trajectory in Table IV. The results of SGAN and SRLSTM with their original batch size (64 and 4, respectively) are also reported. Our approach is 8.04x faster than SGAN with batch size of 1.

V. CONCLUSIONS AND FUTURE WORK

We developed GAMMA, a general motion prediction model that models heterogeneous traffic agents with different physical and behavioral constraints. We proposed a unified framework of constrained geometry optimization that incorporates key factors for motion prediction including kinematics, geometry, intention, attention, and responsibility. Experimental results show that GAMMA significantly outperforms state-of-the-art approaches on various real-world datasets in terms of both prediction accuracy and efficiency.

Currently, we only used simple models for intention, attention, and responsibility. We plan to investigate more sophisticated models for them in order to handle complex scenarios. Note that the GAMMA framework can be easily extended to incorporate more factors, such as courtesy, patience, social comfort zones, etc..

REFERENCES

- [1] J. Van Den Berg, S. Guy, M. Lin, and D. Manocha, "Reciprocal n-body collision avoidance," in *Proc. Int. Symp. on Robotics Research*, 2009.
- [2] D. Helbing and P. Molnar, "Social force model for pedestrian dynamics," *Physical review E*, vol. 51, p. 4282, 1995.
- [3] R. Löhner, "On the modeling of pedestrian motion," *Applied Mathematical Modelling*, vol. 34, pp. 366–382, 2010.
- [4] G. Ferrer, A. Garrell, and A. Sanfeliu, "Robot companion: A social-force based approach with human awareness-navigation in crowded environments," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots & Systems*, 2013.
- [5] S. Pellegrini, A. Ess, K. Schindler, and L. Van Gool, "You'll never walk alone: Modeling social behavior for multi-target tracking," in *Computer Vision, 2009 IEEE 12th International Conference on*, pp. 261–268, IEEE, 2009.
- [6] K. Yamaguchi, A. C. Berg, L. E. Ortiz, and T. L. Berg, "Who are you with and where are you going?," in *Proc. IEEE Conf. on Computer Vision & Pattern Recognition*, pp. 1345–1352, IEEE, 2011.
- [7] P. Fiorini and Z. Shiller, "Motion planning in dynamic environments using velocity obstacles," *Int. J. Robotics Research*, vol. 17, pp. 760–772, 1998.
- [8] J. Van den Berg, M. Lin, and D. Manocha, "Reciprocal velocity obstacles for real-time multi-agent navigation," in *Proc. IEEE Int. Conf. on Robotics & Automation*, 2008.
- [9] J. Snape, J. Van Den Berg, S. J. Guy, and D. Manocha, "The hybrid reciprocal velocity obstacle," *IEEE Transactions on Robotics*, vol. 27, pp. 696–706, 2011.
- [10] N. Deo, A. Rangesh, and M. M. Trivedi, "How would surround vehicles move? a unified framework for maneuver classification and motion prediction," *IEEE Trans. on Intelligent Vehicles*, vol. 3, no. 2, pp. 129–140, 2018.
- [11] A. Houenou, P. Bonenfant, V. Cherfaoui, and W. Yao, "Vehicle trajectory prediction based on motion model and maneuver recognition," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots & Systems*, pp. 4363–4369, IEEE, 2013.
- [12] M. Schreier, V. Willert, and J. Adamy, "Bayesian, maneuver-based, long-term trajectory prediction and criticality assessment for driver assistance systems," in *17th Int. IEEE Conf. on Intelligent Transportation Systems*, pp. 334–341, IEEE, 2014.
- [13] M. Meghiani, Y. Luo, Q. H. Ho, P. Cai, S. Verma, D. Rus, and D. Hsu, "Context and intention aware planning for urban driving," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots & Systems*, 2019.
- [14] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," *arXiv preprint arXiv:1711.03938*, 2017.
- [15] P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, and E. Wießner, "Microscopic traffic simulation using sumo," in *21st Int. Conf. Intelligent Transportation Systems*, pp. 2575–2582, IEEE, 2018.
- [16] C. L. Azevedo, N. M. Deshmukh, B. Marimuthu, S. Oh, K. Marcuk, H. Soh, K. Basak, T. Toledo, L.-S. Peh, and M. E. Ben-Akiva, "Sim-Mobility short-term: An integrated microscopic mobility simulator," *Transportation Research Record*, vol. 2622, no. 1, pp. 13–23, 2017.
- [17] J. Alonso-Mora, A. Breitenmoser, M. Rufli, P. Beardsley, and R. Siegwart, "Optimal reciprocal collision avoidance for multiple non-holonomic robots," in *Distributed Autonomous Robotic Systems*, pp. 203–216, Springer, 2013.
- [18] J. Alonso-Mora, A. Breitenmoser, P. Beardsley, and R. Siegwart, "Reciprocal collision avoidance for multiple car-like robots," in *Proc. IEEE Int. Conf. on Robotics & Automation*, 2012.
- [19] J. Van Den Berg, J. Snape, S. J. Guy, and D. Manocha, "Reciprocal collision avoidance with acceleration-velocity obstacles," in *Proc. IEEE Int. Conf. on Robotics & Automation*, pp. 3475–3482, IEEE, 2011.
- [20] D. Wilkie, J. Van Den Berg, and D. Manocha, "Generalized velocity obstacles," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots & Systems*, pp. 5573–5578, IEEE, 2009.
- [21] A. Best, S. Narang, and D. Manocha, "Real-time reciprocal collision avoidance with elliptical agents," in *Proc. IEEE Int. Conf. on Robotics & Automation*, pp. 298–305, IEEE, 2016.
- [22] S. A. Stüvel, N. Magnenat-Thalmann, D. Thalmann, A. F. van der Stappen, and A. Egges, "Torso crowds," *IEEE Trans. on Visualization and Computer Graphics*, vol. 23, no. 7, pp. 1823–1837, 2017.
- [23] Y. Ma, D. Manocha, and W. Wang, "Efficient reciprocal collision avoidance between heterogeneous agents using ctmat," *Proc. of the 17th Int. Conf. on Autonomous Agents and Multiagent Systems*, 2018.
- [24] Y. Luo, P. Cai, A. Bera, D. Hsu, W. S. Lee, and D. Manocha, "PORCA: Modeling and planning for autonomous driving among many pedestrians," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3418–3425, 2018.
- [25] E. Cheung, A. Bera, and D. Manocha, "Efficient and safe vehicle navigation based on driver behavior classification," in *Proc. IEEE Conf. on Computer Vision & Pattern Recognition*, pp. 1024–1031, 2018.
- [26] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social LSTM: Human trajectory prediction in crowded spaces," in *Proc. IEEE Conf. on Computer Vision & Pattern Recognition*, pp. 961–971, 2016.
- [27] A. Gupta, J. Johnson, L. Fei-Fei, S. Savarese, and A. Alahi, "Social GAN: Socially acceptable trajectories with generative adversarial networks," in *Proc. IEEE Conf. on Computer Vision & Pattern Recognition*, pp. 2255–2264, 2018.
- [28] A. Sadeghian, V. Kosaraju, A. Sadeghian, N. Hirose, and S. Savarese, "SoPhie: An attentive gan for predicting paths compliant to social and physical constraints," *arXiv preprint arXiv:1806.01482*, 2018.
- [29] P. Zhang, W. Ouyang, P. Zhang, J. Xue, and N. Zheng, "SR-LSTM: State refinement for LSTM towards pedestrian trajectory prediction," in *Proc. IEEE Conf. on Computer Vision & Pattern Recognition*, pp. 12085–12094, 2019.
- [30] F. Althché and A. de La Fortelle, "An LSTM network for highway trajectory prediction," in *20th Int. IEEE Conf. on Intelligent Transportation Systems*, pp. 353–359, IEEE, 2017.
- [31] N. Deo and M. M. Trivedi, "Convolutional social pooling for vehicle trajectory prediction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 1468–1476, 2018.
- [32] N. Lee, W. Choi, P. Vernaza, C. B. Choy, P. H. Torr, and M. Chandraker, "DESIRE: Distant future prediction in dynamic scenes with interacting agents," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 336–345, 2017.
- [33] B. Kim, C. M. Kang, J. Kim, S. H. Lee, C. C. Chung, and J. W. Choi, "Probabilistic vehicle trajectory prediction over occupancy grid map via recurrent neural network," in *20th IEEE Int. Conf. on Intelligent Transportation Systems*, pp. 399–404, IEEE, 2017.
- [34] R. Chandra, U. Bhattacharya, A. Bera, and D. Manocha, "TraPHic: Trajectory prediction in dense and heterogeneous traffic using weighted interactions," *arXiv preprint arXiv:1812.04767*, 2018.
- [35] Y. Ma, X. Zhu, S. Zhang, R. Yang, W. Wang, and D. Manocha, "TrafficPredict: Trajectory prediction for heterogeneous traffic-agents," in *Proc. AAAI Conf. on Artificial Intelligence*, 2019.
- [36] S. Pellegrini, A. Ess, and L. Van Gool, "Improving data association by joint modeling of pedestrian trajectories and groupings," in *Proc. European Conf. on Computer Vision*, pp. 452–465, Springer, 2010.
- [37] L. Leal-Taixé, M. Fenzi, A. Kuznetsova, B. Rosenhahn, and S. Savarese, "Learning an image-based motion context for multiple people tracking," in *Proc. IEEE Conf. on Computer Vision & Pattern Recognition*, pp. 3542–3549, 2014.
- [38] A. Sadeghian, V. Kosaraju, A. Sadeghian, N. Hirose, H. Rezatofighi, and S. Savarese, "Sophie: An attentive gan for predicting paths compliant to social and physical constraints," in *Proc. IEEE Conf. on Computer Vision & Pattern Recognition*, pp. 1349–1358, 2019.
- [39] T. Zhao, Y. Xu, M. Monfort, W. Choi, C. Baker, Y. Zhao, Y. Wang, and Y. N. Wu, "Multi-agent tensor fusion for contextual trajectory prediction," in *Proc. IEEE Conf. on Computer Vision & Pattern Recognition*, pp. 12126–12134, 2019.
- [40] J. Amirian, J.-B. Hayet, and J. Pettré, "Social ways: Learning multimodal distributions of pedestrian trajectories with GANs," in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition Workshops*, pp. 0–0, 2019.
- [41] X. Chen, Y. Duan, R. Houthoofd, J. Schulman, I. Sutskever, and P. Abbeel, "InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets," in *Advances in Neural Information Processing Systems*, pp. 2172–2180, 2016.