LEAP-O: Learning to Predict Dynamic Obstacles for Safe Trajectory Planning

Binh Nguyen[1,4], Truong X. Nghiem[1], Linh Nguyen[2], Hung M. La[3], and Thang Nguyen[4,⋆]

***Abstract*— Trajectory planning plays a crucial role in autonomous driving and navigation by enabling robots to generate safe paths while minimizing travel costs and avoiding collisions. This paper addresses the issue of predicting dynamic obstacles for safe trajectory planning when prior information is unavailable and detection range is limited. We propose a learning framework using Gaussian Processes (GP) for motion prediction and uncertainty estimation, further enhanced by Recurrent Neural Networks (RNN) for more accurate predictions. In addition, we develop a receding horizon planning method, formulated as a stochastic optimization problem, to ensure safe, collision-free paths with confidence probabilities. Together, these contributions provide a robust framework for adaptive and safe trajectory generation in dynamic environments. Simulations were performed to demonstrate the effectiveness of the proposed strategy, where our approach (combining GP and RNN) outperformed a baseline method that utilized only GP.**

I. Introduction

Trajectory planning has come ubiquitous in various applications, including navigation and autonomous driving, where robots must generate safe paths in real time to meet specific objectives while avoiding collisions and minimizing travel costs. In dynamic environments, uncertainties in the current states of objects can lead to suboptimal decisions, particularly due to the momentum and inertia of both robots and obstacles. Consequently, predicting the future movements of obstacles becomes essential for effective robot planning. However, the complex and aggressive motions of objects, such as pedestrians and human-driven vehicles, pose challenges for traditional physics-based models like differential equations, which struggle to accurately capture such dynamics. Additional challenges include mismatches between theoretical and actual behavior, noisy sensor data, and interactions among multiple objects. These limitations highlight the need for data-driven models that can account for these complexities.

Given the inherent difficulties in predicting obstacle movements with complete accuracy, uncertainties must be incorporated into the predictions. Prior research on dynamic obstacle avoidance has primarily followed deterministic or probabilistic approaches [1]–[5]. While some studies attempt to approximate obstacle trajectories and determine occupied regions [1], [2], others focus on probabilistic collision avoidance [4], [5], where robots navigate with a certain success rate based on confidence probabilities. The probabilistic models quantify prediction uncertainties through distributions of predicted obstacle movements.

This paper addresses the challenge of predicting dynamic obstacles for safe trajectory planning when there is no prior knowledge of the obstacles and the detection range is limited. Our contributions are twofold:

- **A learning framework for motion prediction of obstacles with unknown dynamics.** Our work takes advantage of GP to predict future movements and quantify uncertainties in moving obstacles. An RNN is responsible for improving the predicted results through GP regression.
- **Optimal safe trajectory planning for collision avoidance.** With informed predictions of obstacle movements, a receding horizon planning formulated in terms of a stochastic optimal problem is used to predict collision-free trajectories for given confidence probabilities.

II. Related works

In recent years, receding horizon planning problems have been extensively studied [2], [6]. A common issue identified in these studies is the "freezing movement" problem, where robots may become congested due to large prediction uncertainties over long horizons. One alternative to address this challenge is incorporating model social interactions, which can be learned through reinforcement learningx [7], [8].

Recurrent Neural Networks (RNNs) have emerged as powerful tools for forecasting object motions using time-series data [9], [10]. They have been widely explored for trajectory prediction [11], [12]. However, these neural network-based predictors often lack the capability to quantify prediction uncertainties, leading to potential mismatches between predicted and actual values. Such discrepancies can result in unsafe or inappropriate decision-making.

As mentioned, developing predictive models is a significant challenge in obstacle avoidance. Recent studies [5], [13] have introduced data-driven models for predicting pedestrian

⋆Corresponding author: Thang Nguyen (thang.nguyen@tamucc.edu)

This material is based upon work supported by the National Science Foundation under Award No. 2514584, by the Texas Comprehensive Research Fund program from the Texas A&M University-Corpus Christi Division of Research and Innovation, and by the Army Research Office under Grant Number W911NF-23-1-0186. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Office or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein.

[1]Department of Electrical and Computer Engineering, College of Engineering and Computer Science, University of Central Florida, Orlando, FL 32816, USA

[2]Institute of Innovation, Science and Sustainability, Federation University Australia, Churchill, VIC 3842, Australia

[3]The Advanced Robotics and Automation (ARA) Lab, Department of Computer Science and Engineering, University of Nevada, Reno, NV 89557, USA

[4]Department of Engineering, Texas A&M University–Corpus Christi, Corpus Christi, TX 78412, USA

movements, with uncertainties estimated through Bayesian inference (BI). Other research has utilized GP as predictive models [14], [15]. However, these approaches typically involve manually selecting prior knowledge, such as the mean function in GP or the likelihood function in BI.

Conformal Prediction (CP) [16], [17] offers another statistical method for assessing prediction uncertainty. For example, [18] uses CP to design a receding horizon controller, while [17] combines RNNs for trajectory prediction with CP to validate prediction regions and ensure collision-free movements. Despite its advantages, a major limitation of CP is its need for large datasets, which may not be available in all scenarios, particularly at the initial stages when robots have limited detection capabilities.

Notations: $\|\cdot\|$ stands for 2-norm in Euclidean space; $\mathbb{R}$ denotes the space of real numbers; for any vectors $\mathbf{v}, \mathbf{u} \in \mathbb{R}^n$, $\mathbf{v} \succ \mathbf{u}$ implies all the elements of $\mathbf{v}-\mathbf{u}$ are greater than zero. Notation $\mathbf{E}(\cdot)$ denotes the expectation operator.

III. Problem Formulation

A. Preliminary

Consider the following discrete-time dynamical system of a robot in two-dimensional space:

$$\mathbf{s}_{t+1} = \mathbf{f}(\mathbf{s}_t, \mathbf{u}_t), \tag{1}$$

where the state vector $\mathbf{s}_t = \begin{bmatrix} x_t, & y_t, & \theta_t \end{bmatrix}^\top \in \mathcal{S} \subset \mathbb{R}^3$ consists of the robot position $\mathbf{p}_t = [x_t,\ y_t]^\top$ and its heading angle θ_t at time t, and $\mathbf{u}_t \in \mathcal{U} \subset \mathbb{R}^2$ denotes the control input at time t. The sets $\mathcal{U}$ and $\mathcal{S}$ represent for permissible control inputs and the workspace of the robot. Additionally, $\mathbf{f} : \mathbb{R}^3 \times \mathbb{R}^2 \to \mathbb{R}^3$ is a differentiable function describing the system dynamics. In this paper, the robot shape is considered a polyhedron of the form:

$$\mathcal{P} = \{\mathbf{z} \in \mathbb{R}^2 | A\mathbf{z} \preceq \mathbf{b}\}, \tag{2}$$

where the matrix $A \in \mathbb{R}^{N\times 2}$ and the vector $\mathbf{b} \in \mathbb{R}^N$ are known parameters. The movement of the robot is now seen as that of $\mathcal{P}$ in the plane. Using the rotation matrix $R(\theta_t) = \begin{bmatrix} \cos\theta_t & -\sin\theta_t \\ \sin\theta_t & \cos\theta_t \end{bmatrix}$, the robot shape $\mathcal{M}(\mathbf{s}_t)$ at the robot pose $\mathbf{s}_t$ is given by

$$\mathcal{M}(\mathbf{s}_t) = \{R(\theta_t)\mathbf{z} + \mathbf{p}_t | \mathbf{z} \in \mathcal{P}\}. \tag{3}$$

The robot operates in an environment with a dynamic obstacle whose trajectories are *a priori* unknown (i.e., no prior knowledge about the obstacle dynamics or kinematics is given). Let $\mathbf{p}_t^o = [x_t^o,\ y_t^o]^\top$ be the position of the center of the obstacle $\mathcal{O}$. We assume that the obstacle is represented by a circle with radius c_r, therefore its shape at position $\mathbf{p}_t^o$ is given by

$$\mathcal{O}(\mathbf{p}_t^o) = \{\mathbf{z} \in \mathbb{R}^2 | \|\mathbf{z} - \mathbf{p}_t^o\|_2 \le c_r\}. \tag{4}$$

We can formulate the collision-free condition at time t as $\mathcal{M}(\mathbf{s}_t) \cap \mathcal{O}(\mathbf{p}_t^o) = \emptyset$. It is assumed that the robot can perceive an obstacle within a distance of R_d, that is, when $\|\mathbf{p}_t - \mathbf{p}_t^o\|_2 \le R_d$. It is also noted that we do not consider the localization uncertainty in this paper. However, sensor measurement errors are incorporated into the learning model.

B. Safe Trajectory Planning Problem

To investigate the problem of dynamic obstacle avoidance, we consider basic trajectory planning problems [19], in which a robot aims to follow a given reference path $\{\mathbf{r}_t\}_{t>0}$ while trying to avoid the obstacle. The constrained finite-horizon optimal control problem with collision avoidance is

$$\min_{\mathbf{s}_{i|t}, \mathbf{u}_{i|t}} \sum_{i=0}^{H-1} c_i(\mathbf{s}_{i|t}, \mathbf{u}_{i|t}) \tag{5a}$$

$$\text{s.t.} \mathbf{u}_{i|t} \in \mathcal{U},\ \mathbf{s}_{i|t} \in \mathcal{S}, \tag{5b}$$

$$\mathbf{s}_{i+1|t} = \mathbf{f}(\mathbf{s}_{i|t}, \mathbf{u}_{i|t}),\ \mathbf{s}_{0|t} = \mathbf{s}_t, \tag{5c}$$

$$\mathcal{M}(\mathbf{s}_{i|t}) \cap \mathcal{O}(\mathbf{p}_{t+i}^o) = \emptyset, \tag{5d}$$

for all $i = 0, 1, \ldots, H-1$, where $\mathbf{s}_{i|t}$ and $\mathbf{u}_{i|t}$ denote the predicted states and control inputs at the i-th step ahead, and H is a specified horizon. Here, c_i is a strictly convex function quantifying the costs of tracking errors and control, e.g., $c_i(\mathbf{s}_{i|t}, \mathbf{u}_{i|t}) = (\mathbf{s}_{i|t} - \mathbf{r}_{t+i})^\top Q_i (\mathbf{s}_{i|t} - \mathbf{r}_{t+i}) + w_i \|\mathbf{u}_{i|t}\|_2^2$ with positive positive definite matrices $Q_i \succ 0$ and weighting factors $w_i > 0$. The constraint (5d) represents the collision avoidance condition between the robot and the obstacle for the horizon.

The **major challenge** of the above control problem is the unknown dynamics, hence the future movements, of the obstacle. Without a reliable prediction of the obstacle's movements $\mathbf{p}_{t+i}^o$, including uncertainty quantification, the collision avoidance constraint (5d) cannot be determined, whether deterministically or stochastically. To address this challenge, our goal, which is also our **main contribution**, in this work is to develop an online data-driven method for learning to predict the movement of a dynamic obstacle with unknown dynamics. Furthermore, our method allows uncertainty quantification of the predictions, resulting in collision avoidance chance constraints. These are then approximated by certainty-equivalent constraints, leading to a tractable optimal control problem.

IV. Learning to Predict Obstacle Movements

As mentioned above, our primary target is to predict the future positions of the obstacle $\mathbf{p}_{t+1}^o, \mathbf{p}_{t+2}^o, \ldots, \mathbf{p}_{t+H}^o$ in a specified horizon H, based on past observations. This paper uses GP for motion prediction with uncertainty quantification of a dynamic obstacle. More specifically, the obstacle movement model is given by:

$$\begin{cases} x_{t+1}^o = x_t^o + \delta_t^x, \\ y_{t+1}^o = y_t^o + \delta_t^y, \end{cases} \tag{6}$$

where δ_t^x and δ_t^y are the horizontal and vertical displacements of the obstacle in one time step.

We employ GP regression [20] to learn and predict the displacements δ_t^x and δ_t^x of the obstacle in (6). In particular, each δ_t^x and δ_t^x is modeled by a GP:

$$\delta_t^x \sim \mathcal{GP}(\mu^x, \kappa^x),\ \delta_t^y \sim \mathcal{GP}(\mu^y, \kappa^y), \tag{7}$$

where μ^x and μ^y are mean functions, $\kappa^x = \kappa(t, t'; \boldsymbol{\vartheta}^x)$ and $\kappa^y = \kappa(t, t'; \boldsymbol{\vartheta}^y)$ are covariance functions with hyperparameter vectors $\boldsymbol{\vartheta}^x$ and $\boldsymbol{\vartheta}^y$, respectively.

Upon detection of an obstacle $\mathcal{O}$ within the range R_d, the robot will monitor the positions of the obstacle. Consider a time instant t after the obstacle has been detected. Let z_τ^x and z_τ^y be measured displacements of the obstacle $x_{\tau+1}^o - x_\tau^o$ and $y_{\tau+1}^o - y_\tau^o$. Due to measurement noises, it is assumed that z_τ^x and z_τ^y are noisy observations of δ_τ^x and δ_τ^y, i.e., $z_\tau^x = \delta_\tau^x + w_\tau^x$ and $z_\tau^y = \delta_\tau^y + w_\tau^y$, where w_τ^x and w_τ^y have i.i.d. zero-mean normal distributions with variances σ_x^2 and σ_y^2. Denote the datasets of the observed obstacle movements since its detection until t by $\mathcal{D}_t^x = \{\mathbf{t}, \mathbf{z}^x\}$ and $\mathcal{D}_t^y = \{\mathbf{t}, \mathbf{z}^y\}$, where $\mathbf{z}^x$ and $\mathbf{z}^y$ collect the observed displacements and $\mathbf{t}$ contains the corresponding timestamps. At a future time instant $h > t$, the posterior distribution of the predicted displacement along dimension $a \in \{x, y\}$ at h is a Gaussian distribution with mean and variance given by:

$$\mu^a(h|\mathcal{D}_t^a) = \mu^a(h) + K_{h\mathbf{t}}^a (K_{\mathbf{tt}}^a + \sigma_a^2 I)^{-1} (\mathbf{z}^a - \mu^a(\mathbf{t})), \quad (8)$$

$$v^a(h|\mathcal{D}_t^a) = K_{hh}^a - K_{h\mathbf{t}}^a (K_{\mathbf{tt}}^a + \sigma_a^2 I)^{-1} K_{\mathbf{t}h}^a, \quad (9)$$

where $K_{hh}^a = \kappa^a(h,h)$, $K_{\mathbf{t}h}^a = (K_{h\mathbf{t}}^a)^\top = \kappa^a(\mathbf{t}, h)$ and $K_{\mathbf{tt}}^a = \kappa^a(\mathbf{t}, \mathbf{t})$.

The choice of the covariance function κ^a is a key factor for the inferred distribution of δ_t^a ($a \in \{x, y\}$) and is typically specified using prior knowledge. Everytime the robot gets new measurements of the obstacle, the hyperparameter $\boldsymbol{\vartheta}^a$ and noise variance σ_a^2 are recalculated by minimizing the negative marginal log likelihood:

$$\boldsymbol{\vartheta}_t^a, \sigma_{a,t} = \underset{\boldsymbol{\vartheta}^a, \sigma_a \succ 0}{\operatorname{argmin}} \mathcal{L}(\boldsymbol{\vartheta}^a, \sigma_a) \quad (10)$$

where $\mathcal{L}(\boldsymbol{\vartheta}^a, \sigma_a) = -\mathbf{z}^{a\top} (K_{\mathbf{tt}}^a + \sigma_a^2 I)^{-1} \mathbf{z}^a - \operatorname{logdet} |K_{\mathbf{tt}}^a + \sigma_a^2 I| - \text{const}$. The selection of μ^a plays a vital role in the prediction of obstacle movements in GP regression (8). In the case of the dynamic obstacle $\mathcal{O}$, the measurement z_t^a often take non-constant/time-varying values. Moreover, we do not have prior knowledge about the variations of z_t^a since the obstacle has an unknown trajectory ahead. Thus, only preceding formulating $\mu^a(h)$ by deterministic functions can lead to conservative results and inaccurate predictions. This paper proposes data-dependent mean functions $\mu_t^a(h)$ (as a replacement for $\mu^a(h)$ in (8)) that are established from prescribed functions and recurrent neural networks (RNNs). In addition, to reduce computational costs, instead of fitting two separate neural networks to calculate $\mu_t^x(h)$ and $\mu_t^y(h)$, in this way, most of the network parameters are shared between the two mean functions.

To be specific, let $\boldsymbol{\mu} = [\mu^x, \mu^y]^\top$ be the prescribed mean vector function and dataset $\mathcal{D}_t = \{\mathbf{t}, \mathbf{z}\}$ collected from data point $(\tau, \mathbf{z}_\tau = [z_\tau^x,\ z_\tau^y]^\top)$ since $\tau \le t$. The data-dependent mean function $\boldsymbol{\mu}_t = [\mu_t^x, \mu_t^y]^\top$ is given by:

$$\boldsymbol{\mu}_t(h) = \begin{cases} \text{RNN}_t(q_t^1, q_t^1, \ldots, q_t^d), & n(\mathcal{D}_t) \ge n_0, \\ \boldsymbol{\mu}(h), & \text{otherwise} \end{cases}$$

where $\text{RNN}_t(\cdot)$ and $q_t^1, q_t^1, \ldots, q_t^d$ denotes output and hidden states of the RNN with depth d at time t; $n(\mathcal{D}_t)$ represents number of data in $\mathcal{D}_t$ and n_0 is minimum required data for training. Our goal here is to obtain $\boldsymbol{\mu}_t(t+1), \boldsymbol{\mu}_t(t+2) \ldots, \boldsymbol{\mu}_t(t+H)$ from $\boldsymbol{\mu}(t|\mathcal{D}_{t-1}) = [\mu^x(t|\mathcal{D}_{t-1}^x),\ \mu^y(t|\mathcal{D}_{t-1}^y)]^\top$ (in (8) at $t-1$) and $\mathcal{D}_t$. Obviously, since $\boldsymbol{\mu}_t \equiv \boldsymbol{\mu}$ (if $n(\mathcal{D}_t) < n_0$) the goal is already carried out since $\boldsymbol{\mu}$ is a prescribed function. Otherwise, the recurrent structure of an RNN [21] is the following form:

$$\begin{cases} \mathbf{q}_t^1 = \mathcal{F}(\boldsymbol{\mu}(t|\mathcal{D}_{t-1}), \mathbf{q}_{t-1}^1), \\ \mathbf{q}_t^i = \mathcal{F}(\boldsymbol{\mu}(t|\mathcal{D}_{t-1}), \mathbf{q}_{t-1}^i, \mathbf{q}_t^{i-1}) \text{ for } i = \{2, \ldots, d\}, \\ \boldsymbol{\mu}_t(t+1) = \mathcal{Y}(\mathbf{q}_t^d), \end{cases}$$

where $\mathcal{F}$ is a parameterized function by different types of RNN, e.g., LSTMs [22] and GRU [23]; $\mathcal{Y}$ is a parameterized output function. To compute the next values $\boldsymbol{\mu}_t(t+2), \ldots, \boldsymbol{\mu}_t(t+H)$, a recursive manner is used by sequentially applying $\boldsymbol{\mu}_t(t+1), \ldots, \boldsymbol{\mu}_t(t+H-1)$, respectively.

With the help of the GP regression, the posterior location estimation of obstacles $\mathcal{O}(\mathbf{s}_i^o)$ is carried out. To begin with, take conditional expectation of (6) with respect to dataset $\mathcal{D}_t^x$ and $\mathcal{D}_t^y$. As a result, we have uncertainties propagation based on (8) and (9) in the following form:

$$\bar{a}_{i+1|t}^o = \bar{a}_{i|t}^o + \mu^a(t_i|\mathcal{D}_t^a), \quad (11)$$

$$v_{i+1|t}^a = v_{i|t}^a + v^a(t_i|\mathcal{D}_t^a) + \sigma_{a,t}^2, \quad (12)$$

for all $a \in \{x, y\}$ where $\bar{a}_{i|t}^o = \mathbf{E}(a_{i|t}^o|\mathcal{D}_t^a)$, $\bar{a}_{0|t}^o = a_t^o$, $v_{0|t}^a = 0$, t_i are i-th sampling instant ahead, and $\sigma_{a,t}$ is an solution of (10) at time t. Thus, by assuming that the obstacle's movements are governed by (6) and (7), the i-th step ahead posterior distribution of the horizontal/vertical obstacle's position at time step t is governed by:

$$a_{t+i}^o \sim \mathcal{N}(\bar{a}_{i|t}^o, v_{i|t}^a). \quad (13)$$

As shown in (13), the posterior estimated position of the obstacle at time t follow the normal distribution where the mean value and covariance are calculated by (11) and (12). Obviously, the covariance will be higher for further estimated steps due to uncertainty propagation (12).

V. Safe Trajectory Planning

This section develops a probabilistic method for solving the safe trajectory planning problem (5), described in section III-B, to achieve the robot's prescribed goal as efficiently as possible while avoiding collisions with high confidence, based on informed predictions of the obstacle motion.

A. Collision Avoidance Constraints

Due to uncertainty in predicting the obstacle motion, the deterministic collision avoidance constraint (5d) is unattainable with absolute certainty. Instead, the deterministic constraint is reformulated as a chance constraint

$$\mathbf{Pr}(\mathcal{M}(\mathbf{s}_{i|t}) \cap \mathcal{O}(\mathbf{p}_{t+i}^o) = \emptyset) \ge \gamma_i, \quad (14)$$

in which $0 < \gamma_i < 1$ denotes the associated satisfaction (confidence) probability at i-th step ahead. Intuitively, the higher γ_i is, the better it is, as it increases safety by reducing the chance of a collision. However, keeping the confidence level γ_i constant and high over the horizon is unnecessary because immediate collision avoidance (when i is small) is more critical than collision avoidance further down the road (when i is large). Furthermore, the robot will replan its trajectory at the next time step, when more information on the obstacle is available, resulting in less uncertainty. Therefore, we choose time-varying decaying γ_i, i.e., $\gamma_i \ge$

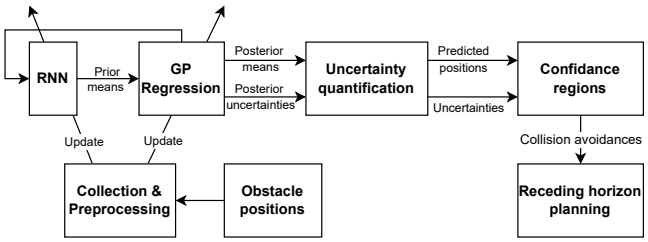

Fig. 1: Scheme of the proposed safe trajectory planning.

γ_{i+1}, to attain a high confidence of immediate collision avoidance but discount the confidence of collision avoidance further into the future.

Let $\bar{\mathbf{p}}^o_{i|t} = \left[\bar{x}^o_{i|t},\ \bar{y}^o_{i|t}\right]^\top$ denote the expected predicted position of the obstacle at future time i. The following proposition presents a sufficient condition for (14).

Proposition 5.1: Consider the robot $\mathcal{M}$ and the obstacle $\mathcal{O}$ as given in (3) and (4), respectively. Define $d^a_{i|t} = \phi^{-1}\left(\frac{1+\sqrt{\gamma_i}}{2}\right)\sqrt{v^a_{i|t}}$ for each $a \in \{x, y\}$, where ϕ is the quantile function of a standard normal distribution [24]. Then, (14) holds if

$$\mathcal{M}(\mathbf{s}_{i|t}) \cap \overline{\mathcal{O}}(\bar{\mathbf{p}}^o_{i|t}) = \emptyset, \tag{15}$$

where $\overline{\mathcal{O}}(\bar{\mathbf{p}}^o_{i|t}) = \mathcal{O}(\bar{\mathbf{p}}^o_{i|t}) \oplus \mathcal{E}_{i|t}$ in which $\mathcal{E}_{i|t} = \mathcal{E}^x_{i|t} \times \mathcal{E}^y_{i|t}$, $\mathcal{E}^a_{i|t} = \{s \in \mathbb{R} || s| \le d^a_{i|t}\}$ for each $a \in \{x, y\}$, $\oplus$ denotes the Minkowski sum of two sets ($A \oplus B = \{a+b | a \in A, b \in B\}$).

Proof: We first show that $\mathbf{Pr}(\mathcal{O}(\mathbf{p}^o_{t+i}) \subset \overline{\mathcal{O}}(\bar{\mathbf{p}}^o_{i|t})) \ge \gamma_i$. From the posterior distribution (13), we have $\mathbf{p}^o_{t+i} = \bar{\mathbf{p}}^o_{i|t} + \mathbf{e}_{i|t}$, where $\mathbf{e}_{i|t} = [e^x_{i|t},\ e^y_{i|t}]$ and $e^a_{i|t} \sim \mathcal{N}(0, v^a_{i|t})$, $\forall a \in \{x, y\}$. It follows that $\mathbf{Pr}(\mathcal{O}(\mathbf{p}^o_{t+i}) \subset \overline{\mathcal{O}}(\bar{\mathbf{p}}^o_{i|t})) = \mathbf{Pr}(\mathcal{O}(\bar{\mathbf{p}}^o_{i|t} + \mathbf{e}_{i|t}) \subset \mathcal{O}(\bar{\mathbf{p}}^o_{i|t}) \oplus \mathcal{E}_{i|t}) = \mathbf{Pr}(\mathcal{O}(\bar{\mathbf{p}}^o_{i|t}) + \mathbf{e}_{i|t} \subset \mathcal{O}(\bar{\mathbf{p}}^o_{i|t}) \oplus \mathcal{E}_{i|t}) \ge \mathbf{Pr}(\mathbf{e}_{i|t} \in \mathcal{E}_{i|t})$. Note that $\mathbf{Pr}(\mathbf{e}_{i|t} \in \mathcal{E}_{i|t}) = \mathbf{Pr}(e^x_{i|t} \in \mathcal{E}^x_{i|t})\mathbf{Pr}(e^y_{i|t} \in \mathcal{E}^y_{i|t})$. Following the definition of the quantile function [24], $\mathbf{Pr}(e^x_{i|t} \in \mathcal{E}^x_{i|t}) \ge \sqrt{\gamma_i}$. As a result, $\mathbf{Pr}(\mathcal{O}(\mathbf{p}^o_{t+i}) \subset \overline{\mathcal{O}}(\bar{\mathbf{p}}_{i|t})) \ge \gamma_i$. Then, for each $\mathbf{p}^o_{t+i}$ such that (15) holds, $\mathcal{O}(\mathbf{p}^o_{t+i}) \subset \overline{\mathcal{O}}(\bar{\mathbf{p}}^o_{i|t})$ implies $\mathcal{M}(\mathbf{s}_{i|t}) \cap \mathcal{O}(\mathbf{p}^o_{t+i}) = \emptyset$, therefore $\mathbf{Pr}(\mathcal{M}(\mathbf{s}_{i|t}) \cap \mathcal{O}(\mathbf{p}^o_{t+i}) = \emptyset) \ge \mathbf{Pr}(\mathcal{O}(\mathbf{p}^o_{t+i}) \subset \overline{\mathcal{O}}(\bar{\mathbf{p}}^o_{i|t})) \ge \gamma_i$. ■

Proposition 5.1 plays a main role in the conversion of the chance constraint (14) into a deterministic constraint (15). The proposed approach for safe trajectory planning is illustrated in Fig. 1 with receding horizon planning discussed in the following section.

B. Receding Horizon Trajectory Planning

Based on *Proposition* 5.1, we first establish analytic conditions for the condition (15). Thus, inspired by [25], the following proposition provides a sufficient condition for collision avoidance.

Proposition 5.2: Consider the robot (3) and the obstacle (4). For variables $\boldsymbol{\lambda}, \boldsymbol{\mu} \in \mathbb{R}^4$ and $\alpha_i \in \mathbb{R}$, (15) is equivalent to the following conditions:

$$\boldsymbol{\lambda} \succ 0, \boldsymbol{\mu} \succ 0, \alpha_i > 0,\ \|\boldsymbol{\mu}^\top E\| \le 1, \tag{16}$$

$$\boldsymbol{\lambda}^\top A + \boldsymbol{\mu}^\top E R(\theta_{i|t}) = 0, \tag{17}$$

$$\boldsymbol{\mu}^\top (E(\mathbf{p}_{i|t} - \bar{\mathbf{p}}^o_{i|t}) - \mathbf{d}_{i|t}) - \boldsymbol{\lambda}^\top \mathbf{b} \ge c_r + \alpha_i, \tag{18}$$

Algorithm 1 LEAP-O

1: **Initialize:** Setup a RNN and a GP.
2: **loop**
3: Collect positions of detected obstacle $\mathcal{O}$.
4: Compute measurement values z^x_t and z^y_t.
5: Update hyperparameters of GP by (10).
6: Update parameters of the RNN.
7: Predict values of mean functions μ^x_t and μ^y_t by the RNN at $t+1, \ldots, t+H$.
8: Compute the posterior distribution (8) and (9).
9: Run uncertainty propagation (11) and (12).
10: Determine an optimal trajectory by NLP (19a).
11: **end loop**

where $\mathbf{d}_{i|t} = \left[d^x_{i|t},\ d^x_{i|t},\ d^y_{i|t},\ d^y_{i|t}\right]^\top$ are defined in *Proposition* 5.1 and $E = \begin{bmatrix} 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & -1 \end{bmatrix}^\top$.

Proof: The proof can be derived using the same approach as in [25, Proposition 3]. ■

It can be seen from the proof of *Proposition* 5.2 that α_i represents for a lower bound of the distance between $\mathcal{M}(\mathbf{s}_{i|t})$ and $\overline{\mathcal{O}}(\bar{\mathbf{p}}^o_{i|t})$. To guarantee no collision in two consecutive sampling time, α_i should be chosen as $\alpha_i \ge v_{\max}\tau$ where $v_{\max}$ stands for the maximum speed of the robot.

Based on optimization (5a) and *Proposition* 5.2, the receding horizon dynamic obstacle avoidance problem is formulated into nonlinear programming problem (NLP):

$$\min_{\mathbf{s}_{i|t}, \mathbf{u}_{i|t}, \boldsymbol{\lambda}_{i|t}, \boldsymbol{\mu}_{i|t}} \sum_{i=1}^{H} c_i(\mathbf{s}_{i|t}, \mathbf{u}_{i|t}) + \ell(\alpha_i) \tag{19a}$$

$$\mathbf{u}_{i|t} \in \mathcal{U},\ \mathbf{s}_{0|t} = \mathbf{s}_k, \tag{19b}$$

$$\mathbf{s}_{i+1|t} = \mathbf{f}(\mathbf{s}_{i|t}, \mathbf{u}_{i|t}) \in \mathcal{S}, \tag{19c}$$

$$\boldsymbol{\lambda}_{i|t} \succ 0, \boldsymbol{\mu}_{i|t} \succ 0, \alpha_i \ge v_{\max}\tau, \|\boldsymbol{\mu}^\top_{i|t} E\| \le 1, \tag{19d}$$

$$\boldsymbol{\mu}^\top_{i|t}(E(\mathbf{p}_{i|t} - \bar{\mathbf{p}}^o_{i|t}) - \mathbf{d}_{i|t}) - \boldsymbol{\lambda}^\top_{i|t}\mathbf{b} \ge c_r + \alpha_i, \tag{19e}$$

$$\boldsymbol{\lambda}^\top_{i|t} A + \boldsymbol{\mu}^\top_{i|t} E R(\theta_{i|t}) = 0, \tag{19f}$$

for $i = 0, 1, \ldots, H-1$ where $\ell(\alpha_i)$ is a differentiable decreasing function representing collision risks. In the light of the above NLP, we outline our scheme in **Algorithm** 1.

VI. Simulation Results

In order to verify our idea in terms of getting a robot to automatically build an ability to learn movements of a dynamic obstacle given its observations and then predict how the obstacle moves in future horizons so that the robot can avoid collision, we conducted extensive experiments in a 2D synthetic environment. The robot only starts learning movements of an obstacle as soon as the obstacle moves in detection range R_d of the robot. As soon as the obstacle goes out of the detection range R_d, the robot will delete all the recordings of the obstacle information as there is no potential collision in a short horizon.

The simulations were carried out using a personal computer (Intel I9, 64 Gb RAM, GPU RTX4090). Code for the simulation was written in Julia and can be found in `https://github.com/EasternBoy/DDDynObs`

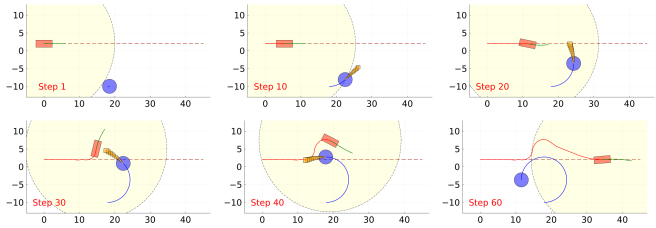

Fig. 2: Scenario 1: A robot (red filled rectangle) is expected to follow a reference path (the dash brown line) while an obstacle (blue filled circle) moves along a circle. The yellow area covered by a dash circle is the detection range of the robot. The red and blue curves are the past trajectories of the robot and obstacle respectively. The green curve is the planned trajectory for the robot to move in near future.

A. Experimental Parameter Settings

To set up the simulations, we defined a robot with a shape as a rectangle mathematically presented in (2), where $A = E$ as proposed in *Proposition 5.2* and $\mathbf{b} = \begin{bmatrix}2.5, & 2.5, & 1.0, & 1.0\end{bmatrix}^\top$. The dynamics of the robot, following [16], is specified by

$$\dot{x} = v\cos\theta,\ \dot{y} = v\sin\theta,\ \dot{\theta} = v\frac{\tan\beta}{\ell},\ \dot{v} = e, \qquad (20)$$

where θ is the yaw angle with respect to horizon axis, $\ell = 1.9$ m is the length between two drive wheels, β is the steering angle, and v and e are the linear velocity and acceleration, respectively. The steering angle was limited to $\pm$ 0.6 rad/s. And v was constrained between -1 and 2 m/s. The discretization of the robot's dynamics was obtained by the forward Euler method as $x_{t+1} = x_t + \tau v_t\cos\theta_t$, $y_{t+1} = y_t + \tau v_t\sin\theta_t$, $\theta_{t+1} = \theta_t + \tau v_t\frac{\tan\alpha_t}{\ell}$ and $v_{t+1} = v_t + \tau e_t$.

As presented by (4), an obstacle in 2D is assumed to be a circle of radius c_r. In the simulations, we set $c_r = 2$ m. Other parameters were also defined as follows: $R_d = 20$ m, $H = 8$, $\tau = 0.1$(s), and $\gamma_i = 0.95$. As presented in Section III-B, in this work we assumed that a robot needs to follow a referred path. And in the simulations, we set that path with trajectories in both directions as $x = 6t$ m and $y = 2$ m, where t is the time variable. The robot's initial position in x direction was set to $x_0 = 0$ m. In this paper we present three different scenarios of obstacle movements.

- **Scenario 1:** An obstacle is moving along a circle with velocity trajectories $\dot{x}^o = 5\cos(\pi t/4) + \omega_x$ and $\dot{y}^o = 5\sin(\pi t/4) + \omega_y$, and its initial position $\mathbf{p}_0^o = [18, -10]^\top$.
- **Scenario 2:** An obstacle is moving along a sine curve with velocity trajectories $\dot{x}^o = -5 + w_x$, $\dot{y}^o = 5\sin(\pi t/4) + w_y$, and its initial position $\mathbf{p}_0^o = [22, 0]^\top$.
- **Scenario 3:** Two obstacles are moving along a circle and a straight line. The first obstacle is in the same setup as Scenario 1, and the second obstacle is moved in $\dot{x}^o = w_x$, $\dot{y}^o = -3.5 + w_y$ with its initial position $\mathbf{p}_0^o = [30, 18]^\top$.

We added i.i.d. zero-mean Gaussian noises, ω_x and ω_y with the same standard deviation of 1, to velocities of the obstacle movements in both directions. In the simulations, the designed RNN has a hidden state size of 16 and a dense layer output of 16 cells.

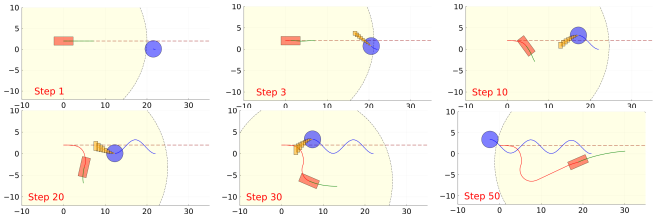

Fig. 3: Scenario 2: All the setups and annotations are similar to Scenario 1 in Fig. 2 except that the obstacle moves along a sine wave.

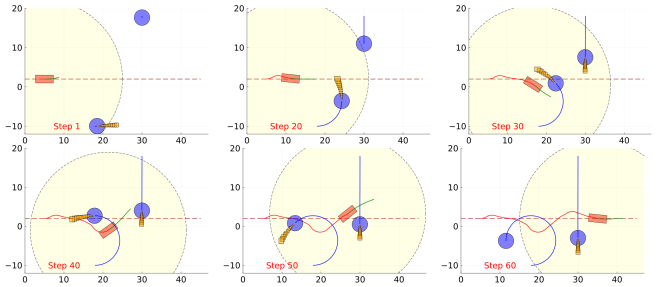

Fig. 4: Scenario 3: All the setups and annotations are similar to Scenario 1 in Fig. 2 except an additional obstacle moving along a vertical line.

B. Performance Discussion

As can be seen from Fig. 2 that at the beginning when the obstacle was outside of the detection range of the robot, the robot easily followed the reference path. Even at time step 10 when the robot could perceive appearance of the obstacle with few position measurements, the robot still comfortably moved along the predefined path as it predicted that the next movements of the obstacle could not cause any collision. At time step 10, given few observations of the obstacle positions, our proposed approach was still able to accurately predict the obstacle's next movements though the dynamics of the obstacle were completely unknown. As illustrated in Fig. 2, the orange rectangles in front of the obstacle present uncertainties $\mathcal{E}_{i|t}$ of the obstacle's next movement predictions, which are relatively small. Width and length parameters of these uncertainties were computed by the use of *Proposition* 5.1. At time step 20, the robot started detouring from the reference path as it predicted that the obstacle soon crossed the reference path, which might cause collision. As evidenced by illustrations at time steps 30 and 40 in Fig. 2, the obstacle clearly crossed and then moved on the reference path at some time but there was no collision. As soon as the robot found safety in motion, it tracked its trajectory back to the reference path as depicted in time step 65 in Fig. 2.

In Scenario 2, the robot found more challenging to avoid the obstacle since the obstacle changed its direction more frequently. As the obstacle constantly moved along a sine wave with different velocities, the uncertainties in the robot predictions of the obstacle's next movements are slightly higher than those in Scenario 1 as illustrated in Fig. 3. However, by employing our approach, the robot was still able to follow the reference path while successfully avoiding collision, which demonstrate robustness of our algorithm.

TABLE I: ROOT MEAN SQUARE ERRORS (m)

Time step	10	20	30	40	50
GP	0.74	1.66	1.06	2.1	0.53
GP+RNN (our method)	0.74	**0.41**	**0.46**	**0.27**	**0.24**

If in both scenarios 1 & 2 the robot was only required to avoid one dynamic obstacle, in Scenario 3 we challenged it further, where two moving obstacles appeared in the same time in front of the robot. This setup simulated more complicated and realistic scenarios in real-world applications in which multiple dynamic obstacles can interfere the robot's traveling path. More specifically, as can be seen in Fig. 4, two moving obstacles intentionally approached the referred path of the robot. It seemed they tried to block the robot's movement. Nevertheless, given our proposed algorithm, the robot could learn and predict the positions of the obstacles very well, which allowed it to not only avoid collision with the obstacles but also insignificantly detour from the reference path. The demonstrations in Fig. 4 obtained in this complex simulation further verify efficacy and robustness of the proposed approach.

To quantify accuracy in predicting the next movements of the obstacle, we computed root mean square errors between the predictions (we only took the predictions of the center of the obstacle as computed by (11)) and the ground truth at several time steps as follows, $\sqrt{\frac{1}{H}\sum_{i}^{H}\|\mathbf{p}^{o}_{t+i} - \bar{\mathbf{p}}^{o}_{i|t}\|^2}$. The obtained results are summarized in Table I. It is noted that for the demonstration purpose, we only present the errors calculated in Scenario 1. For the comparison purpose, we compared our proposed approach (using GP and RNN) with a method (using only GP with a constant mean). As can be seen in Table I, the predictions obtained by our algorithm are much more accurate as a result of simultaneously learning both RNN and GP models.

VII. Conclusions

This paper has proposed a learning framework for motion prediction of obstacles with unknown dynamics, which is based on GP and RNN. In addition, a receding horizon planning method was developed as a stochastic optimization problem to ensure safe, collision-free paths with confidence probabilities. Simulation results illustrated the effectiveness of the proposed strategy, where our approach (combining GP and RNN) outperformed a baseline method that utilized only GP. Numerical experiments of this study focus on specific types of motion patterns, such as circular or sinusoidal movements. For near future work, our work will extend to in real-world obstacles which may exhibit more diverse and unpredictable behaviors, such as abrupt changes in direction, acceleration, or deceleration.

References

[1] P. Ogren and N. E. Leonard, "A convergent dynamic window approach to obstacle avoidance," *IEEE Transactions on Robotics*, vol. 21, no. 2, pp. 188–195, 2005.

[2] S. X. Wei, A. Dixit, S. Tomar, and J. W. Burdick, "Moving obstacle avoidance: A data-driven risk-aware approach," *IEEE Control Systems Letters*, vol. 7, pp. 289–294, 2022.

[3] T. B. Nguyen, T. Nguyen, T. Nghiem, L. Nguyen, J. Baca, P. Rangel, and H.-K. Song, "Collision-free minimum-time trajectory planning for multiple vehicles based on admm," in *2022 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pp. 13785–13790, IEEE, 2022.

[4] C. Fulgenzi, C. Tay, A. Spalanzani, and C. Laugier, "Probabilistic navigation in dynamic environment using rapidly-exploring random trees and gaussian processes," in *2008 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 1056–1062, IEEE, 2008.

[5] D. Fridovich-Keil, A. Bajcsy, J. F. Fisac, S. L. Herbert, S. Wang, A. D. Dragan, and C. J. Tomlin, "Confidence-aware motion prediction for real-time collision avoidance1," *The Int. J. of Robotics Research*, vol. 39, no. 2-3, pp. 250–265, 2020.

[6] A. Thomas, F. Mastrogiovanni, and M. Baglietto, "Probabilistic collision constraint for motion planning in dynamic environments," in *Int. Conf. on Intelligent Autonomous Systems*, pp. 141–154, Springer, 2021.

[7] H. Kretzschmar, M. Spies, C. Sprunk, and W. Burgard, "Socially compliant mobile robot navigation via inverse reinforcement learning," *The Int. J. of Robotics Research*, vol. 35, no. 11, pp. 1289–1307, 2016.

[8] M. Everett, Y. F. Chen, and J. P. How, "Collision avoidance in pedestrian-rich environments with deep reinforcement learning," *Ieee Access*, vol. 9, pp. 10357–10377, 2021.

[9] A. Rudenko, L. Palmieri, M. Herman, K. M. Kitani, D. M. Gavrila, and K. O. Arras, "Human motion trajectory prediction: A survey," *The Int. J. of Robotics Research*, vol. 39, no. 8, pp. 895–935, 2020.

[10] H. Hewamalage, C. Bergmeir, and K. Bandara, "Recurrent neural networks for time series forecasting: Current status and future directions," *Int. J. of Forecasting*, vol. 37, no. 1, pp. 388–427, 2021.

[11] 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 *2017 IEEE 20Th Int. Conf. on intelligent transportation systems (ITSC)*, pp. 399–404, IEEE, 2017.

[12] E. M. Rella, J.-N. Zaech, A. Liniger, and L. Van Gool, "Decoder fusion rnn: Context and interaction aware decoders for trajectory prediction," in *2021 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pp. 5937–5943, IEEE, 2021.

[13] J. F. Fisac, A. Bajcsy, S. L. Herbert, D. Fridovich-Keil, S. Wang, C. J. Tomlin, and A. D. Dragan, "Probabilistically safe robot planning with confidence-based human predictions," *arXiv preprint arXiv:1806.00109*, 2018.

[14] S. Choi, E. Kim, K. Lee, and S. Oh, "Real-time nonparametric reactive navigation of mobile robots in dynamic environments," *Robotics and Autonomous Systems*, vol. 91, pp. 11–24, 2017.

[15] M. Omainska, J. Yamauchi, T. Beckers, T. Hatanaka, S. Hirche, and M. Fujita, "Gaussian process-based visual pursuit control with unknown target motion learning in three dimensions," *SICE Journal of Control, Measurement, and System Integration*, vol. 14, no. 1, pp. 116–127, 2021.

[16] R. Tumu, L. Lindemann, T. Nghiem, and R. Mangharam, "Physics constrained motion prediction with uncertainty quantification," in *2023 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1–8, IEEE, 2023.

[17] L. Lindemann, M. Cleaveland, G. Shim, and G. J. Pappas, "Safe planning in dynamic environments using conformal prediction," *IEEE Robotics and Automation Letters*, 2023.

[18] Y. Chen, U. Rosolia, C. Fan, A. Ames, and R. Murray, "Reactive motion planning with probabilisticsafety guarantees," in *Conf. on robot learning*, pp. 1958–1970, PMLR, 2021.

[19] B. Alrifaee, M. G. Mamaghani, and D. Abel, "Centralized non-convex model predictive control for cooperative collision avoidance of networked vehicles," in *2014 IEEE Int. symposium on intelligent control (ISIC)*, pp. 1583–1588, IEEE, 2014.

[20] C. K. Williams and C. E. Rasmussen, *Gaussian processes for machine learning*, vol. 2. MIT press Cambridge, MA, 2006.

[21] S. Grossberg, "Recurrent neural networks," *Scholarpedia*, vol. 8, no. 2, p. 1888, 2013.

[22] S. Hochreiter, "Long short-term memory," *Neural Computation MIT-Press*, 1997.

[23] R. Dey and F. M. Salem, "Gate-variants of gated recurrent unit (gru) neural networks," in *2017 IEEE 60th Int. midwest symposium on circuits and systems (MWSCAS)*, pp. 1597–1600, IEEE, 2017.

[24] W. Gilchrist, *Statistical modelling with quantile functions*. Chapman and Hall/CRC, 2000.

[25] X. Zhang, A. Liniger, and F. Borrelli, "Optimization-based collision avoidance," *IEEE Transactions on Control Systems Technology*, vol. 29, no. 3, pp. 972–983, 2020.