2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

Robot Trajectron: Trajectory Prediction-based Shared Control for Robot Manipulation

Pinhao Song[1], Pengteng Li[4], Erwin Aertbeliën[1,2], Renaud Detry[1,3]

***Abstract*— We address the problem of (a) predicting the trajectory of an arm reaching motion, based on a few seconds of the motion's onset, and (b) leveraging this predictor to facilitate shared-control manipulation tasks, by reducing the operator's cognitive load through assistance in their anticipated direction of motion. Our novel intent estimator, dubbed the *Robot Trajectron* (RT), produces a probabilistic representation of the robot's anticipated trajectory based on its recent position, velocity and acceleration history. By taking arm dynamics into account, RT can capture the operator's intent better than other SOTA models that only use the arm's position, making it particularly well-suited to assist in tasks where the operator's intent is susceptible to change. We derive a novel shared-control solution that combines RT's predictive capacity to a representation of the locations of potential reaching targets. Our experiments demonstrate RT's effectiveness in both intent estimation and shared-control tasks. We will make the code and data supporting our experiments publicly available at https://gitlab.kuleuven.be/detry-lab/public/robot-trajectron**

I. INTRODUCTION

As robotic hardware improves, teleoperated robot applications emerge at an increasing rate, in domains as varied as subsea maintenance, surgery, or assistive devices. While simple, direct teleoperation is at times feasible, integrators often prefer a form of shared control, where a human operator and an autonomous agent work in tandem, reducing the cognitive load of the operator, and/or improving safety or performance by filtering operator noise and exploiting sensor feedback at a rate that surpasses human capacity. For instance, in robot manipulation controlled by brain-computer interfaces (BCI), the inherent noise in brain signals leads to considerable effort on the part of the patient to realize precise manipulation. With shared control, the user can achieve their goal with increased smoothness and eased effort.

Anticipating the user's intended motion during execution is a crucial component of the shared control paradigm. This ability is usually referred to as *intent estimation*. Current intent estimators assume that the user has a predefined goal and maintains a consistent intent while taking actions to achieve that goal [1], [2], [3], which does not always hold true. Furthermore, most intent estimators rely on position-based methods, which consider only the distance between gripper *position* (past or predicted) and each goal to infer the user's intent [4], [1], ignoring the robot dynamics. For example, position-based MaxEnt IOC like [5], [2], [3] assumes that the user approximately optimizes a cost function which is the cumulative distance between the robot and the goal, and infers the user's intent based on this assumption.

This paper addresses the aforementioned challenges through two primary contributions: First, we propose *Robot Trajectron* (RT), a model that anticipates the trajectory of a robot's end-effector during a reach-to-grasp motion, i.e., RT predicts the end-effector's *future trajectory*. RT bases its prediction on the motion's recent dynamics (position, velocity, and acceleration in the past few seconds), by contrast to prior works that only consider the positions of waypoints of the arm's recent motion [4], [1]. Our model is data-driven, and it learns the robot behavior with few strict assumptions. This characteristic allows it to make predictions with short-term historical dynamics while maintaining noise resiliency, resulting in a fast response to a change of intent, which contrasts with prior work that assumes a fixed intent [5], [2], [3].

Second, we also propose a novel shared-control paradigm that leverages RT as an intent estimator. Our shared-control paradigm follows the basic idea of Artificial Potential Fields (APFs) [6] to guide the robot towards its goals. To flexibly balance RT's vs. the user's, we propose a straightforward agreement mechanism that reinforces the RT's authority in cases of consensus, yet permits the user to override in case of conflict. To assess the efficacy of the approach, comprehensive experiments are conducted with both simulated data and real-world teleoperation tasks. We show that the proposed shared-control paradigm outperforms the prevalent MaxEnt IOC, especially in the case of intent change.

To summarize, the main contributions of this paper are:

- A trajectory prediction model Robot Trajectron, which considers the dynamics of the robot's motion and outputs a probabilistic representation of its future motion. In addition, in the specific case of a tabletop scenario, we provide a means of mapping RT's prediction onto the objects that stand on the table.
- A novel shared-control paradigm is proposed to assist the operator in approaching a goal (i.e., an object) that lies near the predicted trajectory.
- Comprehensive experimental validations are conducted in both a simulation and a real-world grasping task to

Supported by Interne Fondsen KU Leuven/Internal Funds KU Leuven. Partially supported by Flanders Make (strategic research centre for the manufacturing industry).

[1]KU Leuven, Dept. Mechanical Engineering, Research unit *Robotics, Automation and Mechatronics*, B-3000 Leuven, Belgium. Email: firstname.lastname@kuleuven.be

[2]Flanders Make@KU Leuven.

[3]KU Leuven, Dept. Electrical Engineering, Research unit *Processing Speech and Images*, B-3000 Leuven, Belgium.

[4]Shenzhen University, College of Computer Science and Software Engineering, Shenzhen, China. Email: 2110276192@email.szu.edu.cn

show the effectiveness of our method.

II. RELATED WORK

Trajectory Prediction. Trajectory prediction involves estimating future trajectories based on observed paths, which applies not only to pedestrians [7], [8], [9] but also to vehicles [10], [11]. Recently, deep learning methods, due to their strong ability to model social interactions and agents' momentum, have largely outperformed traditional methods. Recurrent Neural Networks (RNNs) have been explored first due to their ability to process sequential data [12], [13]. However, given past trajectories, there can be numerous potential future trajectories. Early RNN-based methods like Social-LSTM [12] which can only generate a single path, fail to capture the multimodal nature of trajectory prediction, which limits its applicability. To handle this challenge, generative architectures have been introduced into trajectory prediction, including Generative Adversarial Network (GAN) [14] and Conditional Variational Autoencoder (CVAE) [15], [16], [11]. For instance, Trajectron [16] follows the CVAE framework and models future velocities using Gaussian Mixture Models (GMMs). This approach provides an explicit distribution of all possible trajectories, offering practicality and flexibility in various applications.

Trajectron shows promise as a robot manipulation intent estimator, but it faces various challenges. The first challenge stems from dealing with 3D data instead of 2D scenarios for which it was designed. Another challenge lies in utilizing Trajectron's output to understand the intent. To overcome these difficulties, RT works in 3D space, and maps the predicted trajectory to the distribution of potential goals (i.e., objects).

Intent Estimation in Shared Control. To efficiently assist users, it is crucial for the system to comprehend their intent. Early studies [17], [18] suggest that mandating explicit intent specification is inefficient and sometimes unfeasible (e.g., BCI-controlled setting). Consequently, contemporary research places emphasis on harnessing implicit cues such as user commands and environmental sensing to deduce user intent. One prevalent approach is to employ a Hidden Markov Model (HMM) for intent inference, treating intent as the model's latent state [19], [20], [5]. Additionally, Bayesian networks [21], [22] have also been explored for intent estimation. One of the intent estimation milestones is MaxEnt IOC [23], which inspires a lot of shared control works and achieves promising performance in the cluttered environment [24], [2], [3]. MaxEnt IOC assumes that the user is an intent-driven agent who seeks to optimize a cost function which is the exponential of the reward. The prevailing way to design to reward is to use the negative distance between the robot and the goal [5]. The distribution over goals can be inferred from the likelihood which is mapped from the rewards of all previous steps.

However, the methods mentioned above ignore motion dynamics and follow a consistent-intent assumption that does not always stand. In this paper, we build a shared control system based on RT, which provides assistance in reaching the goal along a predicted trajectory and promptly adapts to the intent change by considering dynamic information.

III. ROBOT TRAJECTRON

This section introduces our intent-prediction model *Robot Trajectron* (RT). We consider a scenario where an operator wishes to move a manipulator towards one of multiple objects sitting on a table. RT assumes that the user is in the act of guiding the robot from a starting position (usually a rest position) towards a goal (one of the objects). The model predicts the robot's expected future trajectory based on the trajectory it has followed from its start position to its current position. In addition, the model also produces a map that shows where the trajectory is likely to intersect with the table plane.

A. Model Architecture

RT models the probability distribution of the robot's future trajectory, conditioned on the trajectory it has followed to this point. Let us denote the position, velocity, and acceleration of the gripper at a time t with X^t, $\dot{X}^t$, and $\ddot{X}^t$. Let us also denote by $\boldsymbol{x} = [X^{(1:T_{\text{obs}})}, \dot{X}^{(1:T_{\text{obs}})}, \ddot{X}^{(1:T_{\text{obs}})}] \in \mathbb{R}^{T_{\text{obs}} \times 9}$ the history of position, velocity and acceleration from the beginning of the motion to the current time, and by $\boldsymbol{y} = \dot{X}^{(T_{\text{obs}}+1:T_{\text{obs}}+T)} \in \mathbb{R}^{T \times 3}$ the future velocity.

Our aim is to model $p(\boldsymbol{y}|\boldsymbol{x})$, i.e., future velocities conditioned on past positions, velocities and accelerations. As noted in the literature [25], a simple RNN representation of $p(\boldsymbol{y}|\boldsymbol{x})$ may struggle with multimodal distribution, i.e., cases where multiple future trajectories are compatible with a single past trajectory. Instead, we mimic the CVAE framework [25], [16] and introduce a latent variable $\boldsymbol{r}$, to facilitate the encoding of a low-dimensional, multi-modal representation of trajectory data:

$$p(\boldsymbol{y}|\boldsymbol{x}) = \sum_{\boldsymbol{r}} p_\psi(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{r}) p_\theta(\boldsymbol{r}|\boldsymbol{x}). \tag{1}$$

We encode the probability distributions shown above with neural networks, and tune their parameters to maximize the likelihood of a dataset $(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)})$ by maximizing, per CVAE practice [26], [25], the β-weighted evidence-based lower bound (ELBO):

$$\begin{aligned} \max_{\theta,\psi,\phi} \; & \mathbb{E}_{\boldsymbol{r} \sim q_\phi(\boldsymbol{r}|\boldsymbol{x},\boldsymbol{y})}[\log p_\psi(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{r})] \\ & - \beta D_{KL}(q_\phi(\boldsymbol{r}|\boldsymbol{x}, \boldsymbol{y})||p_\theta(\boldsymbol{r}|\boldsymbol{x})), \end{aligned} \tag{2}$$

where $q_\phi(\boldsymbol{r}|\boldsymbol{x}, \boldsymbol{y})$ approximates $p_\theta(\boldsymbol{r}|\boldsymbol{x})$, and θ, ϕ and ψ denote the learnable parameters of the neural representation underlying p_θ, q_ϕ and p_ψ.

In accordance with the CVAE framework, p_θ, q_ϕ and p_ψ are probability distributions. We model p_θ and q_ϕ with Bernouilli distributions whose parameters are generated with multi-layer perceptrons (MLPs) fed by LSTM trajectory encoders (see Fig. 1). We denote by θ_ℓ and ϕ_ℓ the parameters of the two LSTMs that encode the past and future trajectories respectively, and by θ_{m} and ϕ_{m} the parameters of the two corresponding MLPs, with $\theta = (\theta_\ell, \theta_{\text{m}})$ and $\phi = (\phi_\ell, \phi_{\text{m}})$.

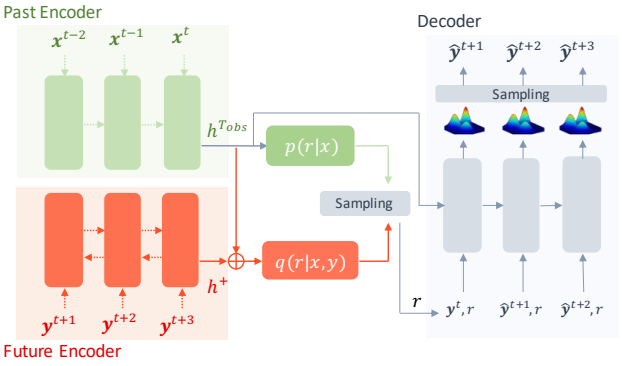

Fig. 1: The architecture of the Robot Trajectron. The red lines denote the train-only operations, while the green lines denote the predict-only operations. See text for details.

Formally, the Bernoulli parameters of p_θ, denoted with B_θ, are obtained with:

$$B_\theta = \text{MLP}(\boldsymbol{h}^{T_{\text{obs}}}; \theta_{\text{m}}), \tag{3}$$

where $\boldsymbol{h}^{T_{\text{obs}}}$ is derived by the past-trajectory LSTM as

$$\boldsymbol{h}^t = \text{LSTM}(\boldsymbol{h}^{t-1}, \boldsymbol{x}^t; \theta_\ell). \tag{4}$$

The Bernouilli parameters of q_ϕ, denoted with B_ϕ, are obtained with $B_\phi = \text{MLP}([\boldsymbol{h}^{T_{\text{obs}}}; \boldsymbol{h}^+]; \phi_{\text{m}})$,where $\boldsymbol{h}^{T_{\text{obs}}}$ is obtained with Eq. 4 and $\boldsymbol{h}^+$ is obtained with the future-trajectory LSTM as $\boldsymbol{h}^t = \text{BiLSTM}(\boldsymbol{h}^{t-1}, \boldsymbol{y}^t; \phi_\ell)$.

We model future velocities p_ψ with velocity-space Gaussian Mixture Models (GMMs) updated at each timestep. We denote the parameters of the GMMs at time t with $G^t = \{(\boldsymbol{\mu}_c^t, \boldsymbol{\Sigma}_c^t, \alpha_c^t)\}_{c=1}^C$, where C is the number of Gaussian components. The decoder models future velocities with GMMs parametrized as follows:

$$[G^t, \boldsymbol{h}^t] = LSTM([\hat{\boldsymbol{y}}^{t-1}, \boldsymbol{r}, \boldsymbol{h}^{t-1}]; \psi), \tag{5}$$

where

$$\boldsymbol{r} \sim \begin{cases} q_\phi(\boldsymbol{r}|\boldsymbol{x}, \boldsymbol{y}), & \text{for training} \\ p_\theta(\boldsymbol{r}|\boldsymbol{x}), & \text{for testing,} \end{cases} \tag{6}$$

and initializing the decoder with $\boldsymbol{h}^{T_{obs}}$. We predict the velocity at time t via sampling, as $\hat{\boldsymbol{y}}^t \sim \text{GMMs}(G^t)$. We note that instead of encoding $\boldsymbol{\Sigma}$ with the six parameters of its matrix representation, we encode it with the six parameters of the lower-triangular matrix $\boldsymbol{L}$ of its Cholesky decomposition $\boldsymbol{\Sigma} = \boldsymbol{L}^T\boldsymbol{L}$. This representation improves training stability and it allows us to effectively sample from a Gaussian with the simple expression $\boldsymbol{\mu} + \boldsymbol{L}\boldsymbol{z}$, $\boldsymbol{z} \sim \mathcal{N}(0, 1)$.

B. Trajectory Prediction

Once trained, RT allows us to obtain the most likely future velocities by sampling from its GMMs, as:

$$\boldsymbol{r}_{\text{best}} = \underset{\boldsymbol{r}}{\text{argmax}}\, p_\theta(\boldsymbol{r}|\boldsymbol{x}), \tag{7}$$

$$\hat{\boldsymbol{y}}_{\text{ml}} = \underset{\boldsymbol{y}}{\text{argmax}}\, p_\psi(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{r}_{\text{best}}). \tag{8}$$

The position trajectory is obtained by integration as:

$$X_{\text{ml}}^t = X_{\text{ml}}^{t-1} + \dot{X}_{\text{ml}}^t \cdot \Delta t, \tag{9}$$

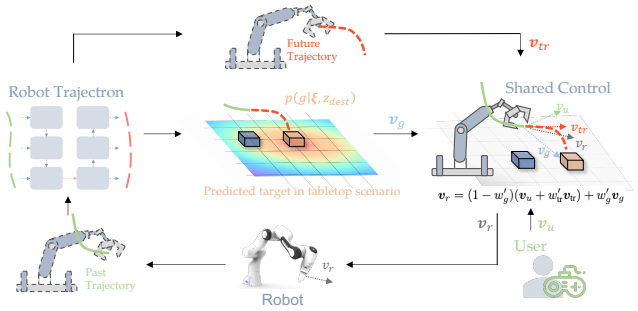

Fig. 2: Overview of the proposed shared control. $\boldsymbol{v}_g$ denotes the velocity towards the most likely object, given the predicted trajectory. $\boldsymbol{v}_{\text{tr}}$ denotes the velocity along the trajectory predicted by RT. $\boldsymbol{v}_u$ denotes noisy user command. See text for details.

where $\hat{\boldsymbol{y}}_{\text{ml}} = \dot{X}_{\text{ml}}^{(T_{\text{obs}}+1:T_{\text{obs}}+T)}$ and Δt is the time interval.

C. Target Selection

The trajectory derived above (9) is conditioned on the robot's motion alone. Motion is however not the only cue that informs intent. An understanding of the robot's environment, obtained through vision for instance, often plays a complementary role. In this section, we assume a tabletop scenario and the availability of the locations of target objects disposed on the table, and we discuss means of combining those object data to trajectory data.

We first use RT to compute a probabilistic representation of locations where the robot's motion is likely to intersect with the table plane. To this end, we first convert the velocity GMMs (5) into position GMMs with:

$$\begin{aligned} \boldsymbol{\mu}_{c,p}^t &= \boldsymbol{\mu}_{c,p}^{t-1} + \boldsymbol{\mu}_c^t \Delta t, \\ \boldsymbol{\Sigma}_{c,p}^t &= \boldsymbol{\Sigma}_{c,p}^{t-1} + \boldsymbol{\Sigma}_c^t (\Delta t)^2, \\ \alpha_{c,p}^t &= \alpha_c^t, \end{aligned} \tag{10}$$

where $\boldsymbol{\mu}_{c,p}^t$, $\boldsymbol{\Sigma}_{c,p}^t$, $\alpha_{c,p}^t$ are the mean, covariance and prior of component c at time t in the position GMMs, which are initialized to the last position in the past trajectory and zero matrices at time T_{obs}, respectively.

To derive which object is intended from the position GMMs in 3D space, we take an intersecting section at the table plane h_{tab} and obtain 2D position GMMs. Given a position of an object denoted as $\boldsymbol{g}$, its probability can be obtained from the 2D position GMMs as:

$$p(\boldsymbol{g}|\boldsymbol{\xi}, h_{\text{tab}}) = \text{2DGMMs}(\boldsymbol{g}|\{G^t\}_{t=T_{\text{obs}}+1}^{T_{\text{end}}}, h_{\text{tab}}) \tag{11}$$

where $\boldsymbol{\xi}$ is the past trajectory, and T_{end} is the end time of the process represented by Eq. (10). The goal with the highest probability will be the intended goal.

IV. Shared Control

In this section, we explain how we use RT to assist the user in a reaching motion. Our method is illustrated in Fig. 2. RT utilizes the robot's motion as input and produces (a) the most likely trajectory and (b) a table-plane GMM representation of likely target points. Considering both outputs from RT and

the user's command, the shared-control system generates a final velocity command to control the robot.

We design our method based on Artificial Potential Fields (APFs) [6], a widely adopted shared-control algorithm, which creates attractive/repulsive fields that guide the motion towards a goal and steer away from obstacles. Our solution uses two attractor fields. First, a Goal Attraction Field (GAF) guides the motion towards the object identified via Eq. (11):

$$U_a^g(\boldsymbol{p}) = w_g \| \boldsymbol{g} - \boldsymbol{p} \|, \tag{12}$$

where $\boldsymbol{g}$ is the position of the goal with the highest probability $p(\boldsymbol{g}|\boldsymbol{\xi}, h_{\text{tab}})$, and $\boldsymbol{p}$ is the position of the robot. w_g is the goal attraction weight, which we calculate as:

$$w_g = \min(\gamma \cdot p(\boldsymbol{g}|\boldsymbol{\xi}, h_{\text{tab}}), \nu), \tag{13}$$

where γ is an amplification coefficient, and ν is a threshold that we set to 0.1 in our experiments. According to Eq. (13), the goal attraction weight is linked to the probability generated by RT, which serves as a valuable indicator of uncertainty. If RT is confident that the current motion points unambiguously towards a certain goal, it significantly influences the motion towards that goal.

Our second attractor field helps the robot along RT's predicted trajectory during segments of the motion where Eq. (11) does not allow us to confidently select a goal. Assuming that $\boldsymbol{p}_{\text{tr}}$ is the first point of the most likely predicted trajectory, we build the Trajectory Following Field (TFF) as:

$$U_a^{\text{tr}}(\boldsymbol{p}) = w_{\text{tr}} \| \boldsymbol{p}_{\text{tr}} - \boldsymbol{p} \|, \tag{14}$$

$$w_{\text{tr}} = \max(\frac{l_{pred}}{l_{past} + l_{pred}}, \zeta), \tag{15}$$

where l_{past} and l_{pred} are the length of the past trajectory and the predicted trajectory, respectively, and ζ is a threshold that we set to 0.7 in our experiments. w_{tr} is the trajectory-following weight. TFF can provide more assistance early on in the reaching motion, reducing the noise and stabilizing the prediction of RT. Combining two fields, we define the robot velocity as:

$$\begin{aligned} \boldsymbol{v}_r &= \boldsymbol{v}_u - (\nabla U_a^g(\boldsymbol{p}) + \nabla U_a^{\text{tr}}(\boldsymbol{p})) \\ &= \boldsymbol{v}_u + w_g \frac{\boldsymbol{g} - \boldsymbol{p}}{\| \boldsymbol{g} - \boldsymbol{p} \|} + w_{\text{tr}} \frac{\boldsymbol{p}_{\text{tr}} - \boldsymbol{p}}{\| \boldsymbol{p}_{\text{tr}} - \boldsymbol{p} \|} \\ &= \boldsymbol{v}_u + w_g \boldsymbol{v}_g + w_{\text{tr}} \boldsymbol{v}_{\text{tr}}, \end{aligned} \tag{16}$$

where $\boldsymbol{v}_g$ and $\boldsymbol{v}_{\text{tr}}$ are the velocities generated by the GAF and TFF, respectively. $\boldsymbol{v}_u$ is the user velocity command, and $\boldsymbol{v}_r$ is the velocity command sent to the robot. $\boldsymbol{v}_r$ considers both the position of the intended goal and the predicted trajectory.

Even though $\boldsymbol{v}_r$ takes RT's uncertainty into account to trade between user commands and AI assistance, the user will still feel a strong impedance if they change their intent (pick a different goal) when the robot is near one of the goals. To address this issue, we propose an agreement mechanism that balances the weight of the user and AI:

$$a_g = \max(\frac{\boldsymbol{v}_u \boldsymbol{v}_g}{\| \boldsymbol{v}_u \| \| \boldsymbol{v}_g \|}, 0), \qquad w_g' = \sqrt{a_g w_g}, \tag{17}$$

$$a_{\text{tr}} = \max(\frac{\boldsymbol{v}_u \boldsymbol{v}_{\text{tr}}}{\| \boldsymbol{v}_u \| \| \boldsymbol{v}_{\text{tr}} \|}, 0), \qquad w_{\text{tr}}' = \sqrt{a_{\text{tr}} w_{\text{tr}}}, \tag{18}$$

where a_g and a_{tr} are the agreement of goal control and trajectory control. The agreement mechanism allows the user to regain authority despite a high RT confidence. Finally, the user command, the GAF and TFF may at times conflict with one another, causing oscillations. We therefore introduce a soft switch control in Eq. (16) as:

$$\boldsymbol{v}_r = (1 - w_g')(\boldsymbol{v}_u + w_{\text{tr}}' \boldsymbol{v}_{\text{tr}}) + w_g' \boldsymbol{v}_g. \tag{19}$$

Accordingly, when the confidence of the goal is low, the robot will tend to follow the user command and the predicted trajectory. When the confidence of the goal is high, the robot will tend to be attracted by the intended goal. With the agreement mechanism, the robot will mainly follow the user's command when it conflicts with the AI.

V. Experiments

To comprehensively evaluate the proposed method, we conducted one evaluation of RT in simulation, one shared-autonomy experiment on a real robot, and a change-of-intent experiment. Our research platform for the experiments is Franka Research 3 with a Microsoft Xbox joystick as the control interface.

A. Experiment in the Simulation Dataset

In this experiment, we aim to demonstrate the performance of RT on a simulation dataset. Following the practice in trajectory prediction [27], [16], we adopted the widely-used evaluation metric Average Displacement Error (ADE) and Final Displacement Error (FDE). ADE calculates the average distance between all the ground-truth and estimated positions in the trajectory, while FDE calculates the distance between the endpoints of ground-truth and predicted trajectories. *Best-of-20* and *Most likely* trajectories are sampled to compute these metrics. To provide a point of reference, we establish a baseline using the *Vanilla Residual LSTM*, which outputs the velocity of each step, with a hidden state size of 128. All the models are trained with Adam optimizer, with a learning rate of 0.001 and batch size of 256. The models are trained on a single RTX 4070Ti GPU.

Data Collection. We collected data with a Franka robot simulated in Pybullet. To collect one trajectory, we randomly sample a reaching target point on the table and move the gripper to approach it along a random approach vector, from a random initial joint configuration. To control the movement, we used the MMC controller [28]. The velocity of the gripper is calculated by:

$$\boldsymbol{v}_e = k({}^0\boldsymbol{T}_e^{-1}\, {}^0\boldsymbol{T}_{e*}), \quad \boldsymbol{v}_e' = \boldsymbol{v}_e + \boldsymbol{z}* \| \boldsymbol{v}_e \|, \tag{20}$$

where k is a gain term, ${}^0\boldsymbol{T}_e \in SE(3)$ is the end-effector instantaneous pose in the robot's base frame, ${}^0\boldsymbol{T}_{e*} \in SE(3)$ is the desired end-effector pose in the robot's base frame, $\boldsymbol{v}_e, \boldsymbol{v}_e' \in \mathbb{R}^3$ are the velocity and the noisy velocity of the end-effector, and $\boldsymbol{z} \in \mathbb{R}^3$ is a noise variable sampled from the uniform distribution $U(-1, 1)$. The noisy velocity will be applied to the end-effector with a frequency of 20Hz. According to Eq. (20), the noise level is dependent on the magnitude of the velocity. Finally, we generate 100,000

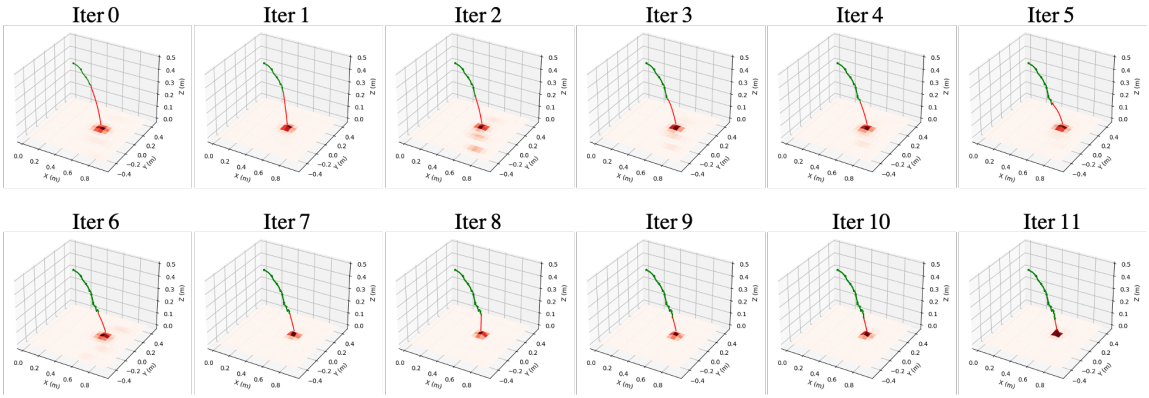

Fig. 3: Visualization of RT. The most likely trajectory and the 2D table GMMs are shown. The green line denotes the past trajectory, while the red line denotes the predicted trajectory.

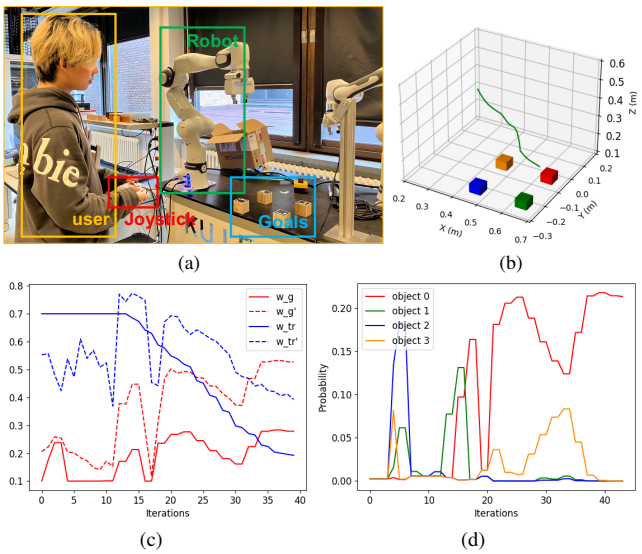

Fig. 4: (a) The set up of the shared autonomy experiment. (b) A demonstration assisted by the proposed method. (c) The weight change in Fig. b. (d) The distribution change of each goal in Fig. b.

TABLE I: The performance comparison in Traj100k.

Method	Best-of-20 (mm)		Most likely (mm)	
	ADE	FDE	ADE	FDE
Vanilla LSTM [29]	-	-	136.95	115.47
Robot Trajectron	17.82	26.97	30.58	49.94

trajectories, among which 90,000 are for training and 10,000 are for testing. We name this dataset as *Traj100k*.

Performance in the Traj100k dataset. Results are shown in Table I. We only report Vanilla LSTM's most likely results since it is a deterministic method. RT performs substantially better than Vanilla LSTM in both *Best-of-20* and *Most likely* settings. The small error of RT demonstrates the strong capability to model the future trajectory.

Visualization of the intended point. Fig. 3 shows an example of trajectory prediction per Eq. (11). Both the most likely trajectory and the 2D GMMs of Eq. (11) are visualized. At the onset of the motion, RT's predicted reaching target is affected by a large uncertainty. In Iter 2, it can be seen that several separated Gaussian components are projected at the table plane, which shows the multi-modal modeling of the user intent. As the gripper moves, the generated distribution is increasingly concentrated, which indicates the increasing confidence in RT's prediction.

TABLE II: Shared autonomy experiment. The total time, the number of inputs and the average sum of the length of 4 trajectories in one round l_{tr} are used as metrics.

Method	Time (sec)	Input	Average l_{tr} (m)
Teleop. [3]	9.36±0.71	41.8±2.8	2.452±0.246
MaxEnt IOC [3]	7.24±0.33	33.8±1.2	2.007±0.060
Robot Trajectron	7.17±0.43	33.8±1.3	1.981±0.092

B. Shared Autonomy Experiment

Design. In this experiment, we will compare our shared control method with two baselines. The first baseline is pure user control (named Teleop.). The other is the prevalent shared control method MaxEnt IOC [24], [2], [3], for which we used the open-source code from the implementation of [3]. In order to make a fair comparison, we use a constant velocity to 0.1 m/s with all methods. The velocity controller MMC [28] is leveraged to control the robot. The experimental setting is shown in Fig. 4a. 4 small cubes equipped with ArUCo markers were placed on the table. In each round, the user was required to sequentially approach the cubes on the table (4 trials) – at which point, in a real-life task, an autonomous grasp controller would take over. User input consists of the direction of the velocity vector, which they can control via six joystick buttons (two buttons for each axis). As written above, the velocity is kept at a constant 0.1 m/s. Three metrics are used for comparison: the total time, the number of inputs (button pushes) and the average sum of the length of 4 trajectories in one round.

Protocol. We enrolled 10 novice participants from the local community. They received training in using our 3-axis joystick to control the robot. During the formal experiment, the order of control methods was randomized for each participant. When the gripper neared a goal, the robot automatically performed the grasping action.

Anaylsis. Table II shows that our proposed method uses less time and fewer inputs and produces shorter trajectories to

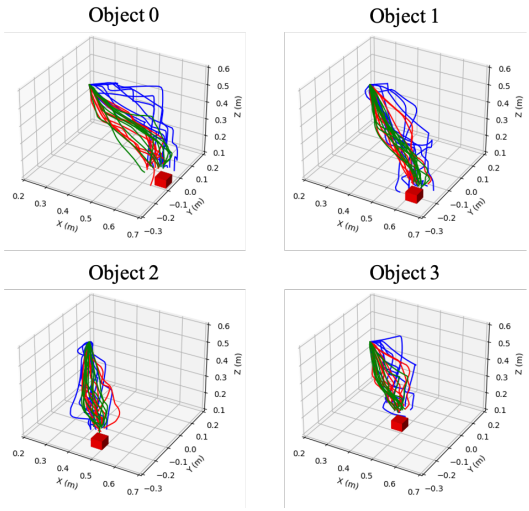

Fig. 5: User demonstrations. The blue lines denote the trajectories fully controlled by the user. The red and green lines denote the trajectories assisted by *MaxEnt IOC* and *Robot Trajectron*, respectively. As discussed in the text, the trajectories guided by RT to reach the object are straighter and smoother.

approach the goal, demonstrating its effectiveness. Notably, our method achieves comparable performance to the SOTA method MaxEnt IOC. The superior performance is due to the early capturing of the robot's motion. Guided by the smooth predicted trajectory, the robot can achieve the goal faster. Fig. 5 shows the trajectories of user demonstrations for 4 cubes. Our method aids in producing smoother and more direct trajectories leading to the goal. Fig. 4b 4c and 4d depict the trajectory, shared weights, and goal distributions within one demonstration. Initially, the user explores the path to the goal with a low GAF weight w_g, providing limited assistance. In this phase, It is mainly TFF that is at work. Although the TFF weight w_{tr} is high, the user input does not align with v_{tr}, leading to a reduction in the agreed TFF weight w'_{tr} due to the agreement mechanism. After the 15th iteration, Trajectron is increasingly confident about object 0, which means that GAF is at work while TFF is ceasing operation. Comparing the original and agreed GAF weights w_g and w'_g, we can see that the agreement mechanism then strengthens the AI's control, resulting in smoother and more efficient gripper movements towards the intended object.

C. Experiment of Intent Estimation

In this final experiment, we evaluate the model behaviors in a situation where the user changes their intent during approaching.

Design. We pre-recorded 10 change-of-intent trajectories with the same object setting as in the shared autonomy experiment. As shown in Fig. 6a, first, the subject will be required to approach one of the objects, and then switch to another. The first part of the motions allows us to measure the robustness of different models to user-input noise, whereas the second part allows us to measure how well a model adapts to a change of intent. We replay these trajectories

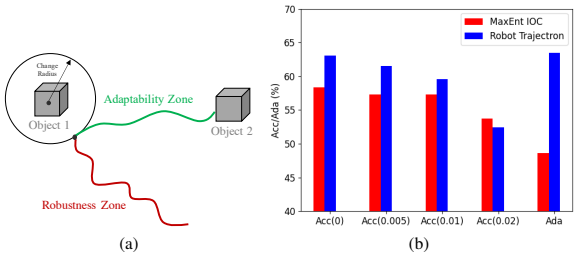

Fig. 6: (a) The illustration of the intent estimation experiment. (b) The results of the intent estimation experiment. "Acc(ϵ)" denotes the Robustness Zone accuracy in different noise levels ϵ. "Ada" denotes the Adaptability.

to both RT and MaxEnt IOC and evaluate their performance in the following metrics.

Metrics. In this experiment, we consider both adaptability and robustness as metrics for they are often conflicting qualities. We evaluate those as follows: (i) **Accuracy**: For each step in the pre-recorded trajectories, the intent estimation model identifies the object with the highest probability as the goal. We evaluate the accuracy of intent estimation by dividing the number of correct predictions by the total number of steps. (ii) **Robustness**: In the *Robustness Zone*, we introduce noise by adding z sampled from a uniform distribution $U(-\epsilon, \epsilon)$ to the trajectories. We then evaluate the accuracy of intent prediction at various noise levels ϵ as an indicator of robustness. (iii) **Adaptability**: A model with good adaptability can quickly perceive the intent change and make more accurate predictions in the *Adaptability Zone*. We measure adaptability by calculating the accuracy of intent prediction in this zone. These metrics help us assess how well the models respond to intent change and their ability to maintain accurate predictions in the presence of noise.

Analysis. The results of the experiment are illustrated in Fig. 6b. From the last column, the Adaptability of our method is higher than MaxEnt IOC, which is due to perceptiveness to the dynamic change. Besides, in the less noisy conditions ($\epsilon \leq 0.01m$), our method still outperforms MaxEnt IOC, because our method can capture the dynamics early on, while MaxEnt IOC still needs to move close enough to the object to make correct predictions. However, when the noise level increases to 0.02m, MaxEnt IOC outperforms our method, for the reason that RT mistakes the high noise for a signal of intent change.

VI. Conclusions

This work addressed limitations related to the consistent-goal assumption of current shared-control works, by proposing a motion predictor that intrinsically captures user intent. By considering motion dynamics, RT can promptly adapt to changes of intent. We combine this predictor to a representation of possible goals to build a potential-field shared control solution. We demonstrated applicability to predicting future motion trajectories, and effectiveness in shared control of a physical robot. In future work, we intend to study the applicability of our work to BCI-controlled grasping.

REFERENCES

[1] Y. Xu, H. Zhang, L. Cao, X. Shu, and D. Zhang, "A shared control strategy for reach and grasp of multiple objects using robot vision and noninvasive brain–computer interface," *IEEE Transactions on Automation Science and Engineering*, vol. 19, no. 1, pp. 360–372, 2020.

[2] K. Muelling, A. Venkatraman, J.-S. Valois, J. E. Downey, J. Weiss, S. Javdani, M. Hebert, A. B. Schwartz, J. L. Collinger, and J. A. Bagnell, "Autonomy infused teleoperation with application to brain computer interface controlled manipulation," *Autonomous Robots*, vol. 41, pp. 1401–1422, 2017.

[3] A. Gottardi, S. Tortora, E. Tosello, and E. Menegatti, "Shared control in robot teleoperation with improved potential fields," *IEEE Transactions on Human-Machine Systems*, vol. 52, no. 3, pp. 410–422, 2022.

[4] Y. Tamura, M. Sugi, J. Ota, and T. Arai, "Prediction of target object based on human hand movement for handing-over between human and self-moving trays," in *ROMAN 2006-The 15th IEEE International Symposium on Robot and Human Interactive Communication*. IEEE, 2006, pp. 189–194.

[5] D. Aarno and D. Kragic, "Motion intention recognition in robot assisted applications," *Robotics and Autonomous Systems*, vol. 56, no. 8, pp. 692–705, 2008.

[6] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *The international journal of robotics research*, vol. 5, no. 1, pp. 90–98, 1986.

[7] J. Yue, D. Manocha, and H. Wang, "Human trajectory prediction via neural social physics," in *European Conference on Computer Vision*. Springer, 2022, pp. 376–394.

[8] A. Vemula, K. Muelling, and J. Oh, "Social attention: Modeling attention in human crowds," in *2018 IEEE international Conference on Robotics and Automation*. IEEE, 2018, pp. 4601–4607.

[9] J. M. Wang, D. J. Fleet, and A. Hertzmann, "Gaussian process dynamical models for human motion," *IEEE transactions on pattern analysis and machine intelligence*, vol. 30, no. 2, pp. 283–298, 2007.

[10] K. Katuwandeniya, S. H. Kiss, L. Shi, and J. V. Miro, "Exact-likelihood user intention estimation for scene-compliant shared-control navigation," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 6437–6443.

[11] T. Salzmann, B. Ivanovic, P. Chakravarty, and M. Pavone, "Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVIII 16*. Springer, 2020, pp. 683–700.

[12] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social lstm: Human trajectory prediction in crowded spaces," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 961–971.

[13] F. Bartoli, G. Lisanti, L. Ballan, and A. Del Bimbo, "Context-aware trajectory prediction," in *2018 24th international conference on pattern recognition (ICPR)*. IEEE, 2018, pp. 1941–1946.

[14] A. Gupta, J. Johnson, L. Fei-Fei, S. Savarese, and A. Alahi, "Social gan: Socially acceptable trajectories with generative adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 2255–2264.

[15] K. Mangalam, H. Girase, S. Agarwal, K.-H. Lee, E. Adeli, J. Malik, and A. Gaidon, "It is not the journey but the destination: Endpoint conditioned trajectory prediction," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*. Springer, 2020, pp. 759–776.

[16] B. Ivanovic and M. Pavone, "The trajectron: Probabilistic multi-agent trajectory modeling with dynamic spatiotemporal graphs," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 2375–2384.

[17] D. Vanhooydonck, E. Demeester, M. Nuttin, and H. Van Brussel, "Shared control for intelligent wheelchairs: an implicit estimation of the user intention," in *Proceedings of the 1st international workshop on advances in service robotics (ASER'03)*, 2003, pp. 176–182.

[18] M. A. Goodrich and D. R. Olsen, "Seven principles of efficient human robot interaction," in *SMC'03 Conference Proceedings. 2003 IEEE International Conference on Systems, Man and Cybernetics. Conference Theme-System Security and Assurance (Cat. No. 03CH37483)*, vol. 4. IEEE, 2003, pp. 3942–3948.

[19] H. Ding, G. Reißig, K. Wijaya, D. Bortot, K. Bengler, and O. Stursberg, "Human arm motion modeling and long-term prediction for safe and efficient human-robot-interaction," in *2011 IEEE International Conference on Robotics and Automation*. IEEE, 2011, pp. 5875–5880.

[20] D. Aarno, S. Ekvall, and D. Kragic, "Adaptive virtual fixtures for machine-assisted teleoperation tasks," in *Proceedings of the 2005 IEEE international conference on robotics and automation*. IEEE, 2005, pp. 1139–1144.

[21] O. C. Schrempf, D. Albrecht, and U. D. Hanebeck, "Tractable probabilistic models for intention recognition based on expert knowledge," in *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2007, pp. 1429–1434.

[22] K. A. Tahboub, "Intelligent human-machine interaction based on dynamic bayesian networks probabilistic intention recognition," *Journal of Intelligent and Robotic Systems*, vol. 45, pp. 31–52, 2006.

[23] B. D. Ziebart, A. L. Maas, J. A. Bagnell, A. K. Dey, *et al.*, "Maximum entropy inverse reinforcement learning." in *Aaai*, vol. 8. Chicago, IL, USA, 2008, pp. 1433–1438.

[24] S. Javdani, H. Admoni, S. Pellegrinelli, S. S. Srinivasa, and J. A. Bagnell, "Shared autonomy via hindsight optimization for teleoperation and teaming," *The International Journal of Robotics Research*, vol. 37, no. 7, pp. 717–742, 2018.

[25] K. Sohn, H. Lee, and X. Yan, "Learning structured output representation using deep conditional generative models," *Advances in neural information processing systems*, vol. 28, 2015.

[26] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner, "beta-vae: Learning basic visual concepts with a constrained variational framework," in *International conference on learning representations*, 2016.

[27] T. Gu, G. Chen, J. Li, C. Lin, Y. Rao, J. Zhou, and J. Lu, "Stochastic trajectory prediction via motion indeterminacy diffusion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 17 113–17 122.

[28] J. Haviland and P. Corke, "A purely-reactive manipulability-maximising motion controller," *arXiv preprint arXiv:2002.11901*, 2020.

[29] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.