An Actor-Critic Approach for Legible Robot Motion Planner

Xuan Zhao[1*], Tingxiang Fan[2*], Dawei Wang[2], Zhe Hu[1], Tao Han[1], Jia Pan[2]

***Abstract*— In human-robot collaboration, it is crucial for the robot to make its intentions clear and predictable to the human partners. Inspired by the mutual learning and adaptation of human partners, we suggest an actor-critic approach for a legible robot motion planner. This approach includes two neural networks and a legibility evaluator: 1) A policy network based on deep reinforcement learning (DRL); 2) A Recurrent Neural Networks (RNNs) based sequence to sequence (Seq2Seq) model as a motion predictor; 3) A legibility evaluator that maps motion to legible reward. Through a series of human-subject experiments, we demonstrate that with a simple handicraft function and no real-human data, our method lead to improved collaborative performance against a baseline method and a non-prediction method.**

I. Introduction

Recent development in robotics and artificial intelligence has brought opportunities for introducing human-robot collaboration into labor-intensive industries such as dexterous assembly. However, it also presents many challenges for researchers to make robots comparable to their human competitors. One such challenge is to build a tacit understanding between the robot and its human partner for smooth human-robot collaboration. The tacit understanding is commonly known as the collaborators ability to predict each others behavior. It is crucial for collaboration since it has been proved as a major factor to influence the human collaborators comfort, safety, and efficiency in the task [1]. To achieve better tacit understanding, it is crucial for the robot to make its intentions clear and predictable to the human partners.

The idea that motions inherently communicate intent is rooted in human perception research. Neuroscience research has suggested that the human brain is designed to detect and extract intention from biological motion [2]. Similarly, robotics research has shown that the manner, in which a robot arm moves as it performs a manipulation task, influences how human interpret and interact with it [3]. To reach well-interpreted and smooth interaction, other works focused on finding a special robot motion type such as *legible motion* [4], *readable motion* [5], and *anticipatory motion* [6].

Inspired by the implicit communication between human workers, we propose a method that emulates the communicating and adapting process by two neural networks, where one network plays the active worker's role and the other plays the passive one. An artificial neural network is a computing system inspired by the biological neural networks that constitute animal brains. Comparing to traditional machine learning method, it shows much capacity to model more complex data and therefore could solve broader problems. However, for a simple artificial neural network, its capability is much limited in comparison with a human brain. Thus, we assume that *if a simple neural network could predict the implicit intent of an active worker's motion very well, a human user can also learn it rapidly and accurately.*

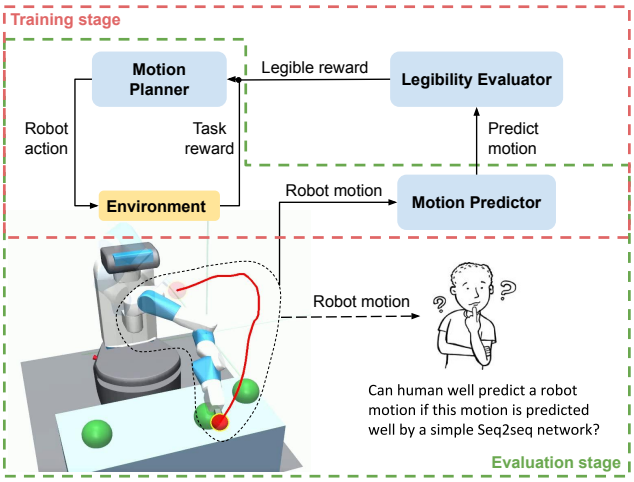

Fig. 1: Our method includes two stages. In the training stage, we train a motion planning policy through an actor-critic approach between motion planner (actor) and motion predictor (critic). The planner tries to generate motion that can be predicted more accurately by the predictor, while the predictor tries to minimize its prediction error. In the evaluation stage, human users are invited to watch the robot motion to evaluate its legibility.

Recently, DRL shows a strong ability in controlling robot arms in manipulation tasks [7] [8], and RNNs commonly performs well in temporal prediction tasks [9]. Based on this, we suggest a joint learning architecture (shown in Figure 1), which includes two neural networks: 1) a policy network trained based on DRL and 2) a motion predictor network based on Seq2Seq. Specifically, the policy network works as a motion planner and the predictor network learns to predict the robot's future motion. Furthermore, following the predicted results, the policy network can optimize its strategy to generate a motion that can be predicted more accurately by the predictor network. In this work, we evolve these two networks together and eventually obtain a legible motion planner that allows the predictor network to predict its generated movements as accurate as possible and as efficient as possible. Our user study supports that considering prediction from a simple neural network improves the legibility and

* denotes equal contribution

[1] X. Zhao, Z. Hu, T. Han are with the Department of Biomedical Engineering, City University of Hong Kong

[2] T. Fan, D. Wang, J. Pan are with the Department of Computer Science, The University of Hong Kong. Corresponding author jpan@cs.hku.hk

This work was partially supported by HKSAR General Research Fund (GRF) HKU 11202119, 11207818

learnability of robot motion.

To summarize, the contribution of this paper includes:

1) Propose an actor-critic joint learning architecture to learn a legible motion planner. This model successfully improves the legibility and learnability of robot motion without real-world data.
2) Propose a legible reward function that can apply to tasks in 3D discrete space.
3) Present a human user evaluation of the motion generated from our method, a baseline method and a non-prediction method.

This paper is organized as follows. After presenting related work in Section II, we present our method to accomplish legible robot motion in Section III. Then we show the experiment design illustrated in Figure 4. Section IV introduces the experiment and two measurements of the experiment. Section V shows the result of two measurements and finally we conclude this paper in Section VI.

II. Related work

Prior research on improving human perceptions of robot motion was primarily based on the *action-to-goal inference* task where a robot needs to conveys its goal with its ongoing motion [4]. Most existing work in this area utilized the *trajectory optimization* technique, where a robot's motion was optimized to achieve specified motion properties, subject to a set of constraints [10]. Some literature suggested a few handicraft motion properties that can improve transfer on robot intent. For example, [4] proposed a heuristic cost function that can generate motions to achieve maximal clarity of a certain goal. This cost function was then tested on both a robot manipulation and a pointing task [11] to show its generability. [12] considered the expressiveness of existing movement approaches, and suggested a combination of "shortest", "straight" and "curved" movement. Similarly, [10] suggested three heuristic functions including "shortest", "straight" and "legible". Besides the work using traditional motion planners, [13] introduced a reinforcement learning method that learns a policy from a user's feedback. In this method, the authors optimize the weights of dynamical movement primitive subject to a cost function includes: user guess efficiency, robustness, and energy. All of these methods use a handicraft function with adjustable or trainable parameters. However, the performance of handicraft functions always depend on cases, and thus they always need to be adjusted for different environments and tasks.

Our model uses DRL to generate robot motion policy and a Seq2Seq predictor to help measure the legibility of robot motion. This data-driven model does not require collecting real human data due to the usage of joint learning. The model could also be trained on different tasks without handcrafting new functions or re-collecting human data.

III. Approach

Our network consists of two parts, one is a reinforcement learning network for generating robot motion; the other is a seq2seq network to predict robot motion. We first describe the modelling of robot motion and the motion's legibility. Then we introduce the architecture of the network and its usage in legible motion planning problem.

A. Trajectory modeling

Inspired by previous work, we consider that a good legible motion in manipulation task can help a user predict a robot's target both quickly and accurately. To numerically calculate the legibility, we first introduce the modeling of robot trajectory, and then the formulation of legibility.

We denote $\boldsymbol{\zeta}_{1:T}$ as the robot trajectory from time-step $t = 1$ to $t = T$. At each time-step, $\boldsymbol{\zeta}_t$ consists of robot joint angles $\boldsymbol{\zeta}_t^J = [\zeta_t^1, \zeta_t^2, \cdots, \zeta_t^P]^\top$ and the position of end-effector $\boldsymbol{\zeta}_t^E = [\zeta_t^x, \zeta_t^y, \zeta_t^z]^\top$, where P denotes the total number of robot joints, J and E denote *joint angles*, *end-effector positions* respectively, and x, y, z are for each dimension of the 3D coordinates. Therefore, the robot trajectory from time-step 1 to t can be represented as: $\boldsymbol{\zeta}_{1:t} = \begin{bmatrix} \boldsymbol{\zeta}_{1:t}^J \\ \boldsymbol{\zeta}_{1:t}^E \end{bmatrix}$.

B. Legibility formulation

Here we suggest three functions to evaluate the legibility of a trajectory. The first one is to finish a task as fast as possible, the second one considers the actual end-effector position, and the last one takes into account the predicted end-effector positions. The details of these three functions are as follows.

1) Fast approaching (*FastApp*): Denote p_{g_i} as the position of the i_{th} goal in goal sets. An eligible action should try to quickly reduce the Euclidean distance $d_t^i = \left\| \zeta_t^E - p_{g_i} \right\|_2$ between the robot end-effector and the robot selected target g_0, or more formally, minimize
$$r_{legible}^t = d_{t-1}^0 - d_t^0.$$
2) End-effector distance (*EffDist*): This is inspired by the *point loss* function from Bodden *et al.*'s work [10]: The predicted goal is the closest one to the end-effector among all the elements in a goal set. Unlike Bodden *et al.*'s work where the end-effector position is first projected to a plane, then the robot moves to the top of the plane and let the projection overlaps with the true goal, our *EffDist* function rewards the action that maximizes the distance dissimilarity of the true goal and other alternative goals. In this way, our function is applicable to cases that the goal set is in 3D space. The formulation for *EffDist* is:
$$r_{legible}^t = e^{\frac{-t}{30}} \min_i \left(\log\left(\frac{\left| d^0 - d^i \right|}{|d^0 + 1|} + 1 \right) \cdot \mathbf{1}_{\{d^0 < d^i\}} \right) \tag{1}$$
where $\mathbf{1}_{\{d^0<d^i\}} = \begin{cases} 1, & d^0 < d^i; \\ -1, & otherwise, \end{cases}$ $e^{-t/30}$ is a penalty coefficient according to time so that a long time duration gives a large penalty.
3) Trajectory distance (*TrajDist*): This is a variation of *EffDist* implying that the predicted goal should be the member of the goal set that is closest to the *predicted*

end-effector positions according to the Euclidean distance. The formulation is similar to Equation 1, except that d^i is changed to $\tilde{d^i} = \min_{\zeta_t^E \in \zeta_{pred}^E} \left\| \zeta_t^E - p_{g_i} \right\|_2$.

C. Robot motion planning

At each time-step t, the robot has access to an observation $\mathbf{o}_t$, and computes a feasible steering command $\mathbf{a}_t$ that drives the robot end-effector to approach the goal $\mathbf{g}$ from the current robot configuration ζ_t. The observation $\mathbf{o}_t$ is drawn from a probability distribution w.r.t. the underling system state $\mathbf{s}_t$, which can be represented as $\mathbf{o}_t \sim \mathbf{O}(s_t)$.

The motion planning problem can be formulated as Partially Observable Markov Decision Process (POMDP) solved with reinforcement learning. Formally, a POMDP can be described as a 6-tuple $(\mathbf{S}, \mathbf{A}, \mathbf{P}, \mathbf{R}, \Omega, \mathbf{O})$, where $\mathbf{S}$ is the state space, $\mathbf{A}$ is the action space, $\mathbf{P}$ is the state-transition model, $\mathbf{R}$ is the reward function, Ω is the observation space ($\mathbf{o} \in \Omega$) and Ω is the observation probability distribution given the system state ($\mathbf{o} \sim \mathbf{O}(s)$). Below we describe the details of the observation space, the action space, and the reward function.

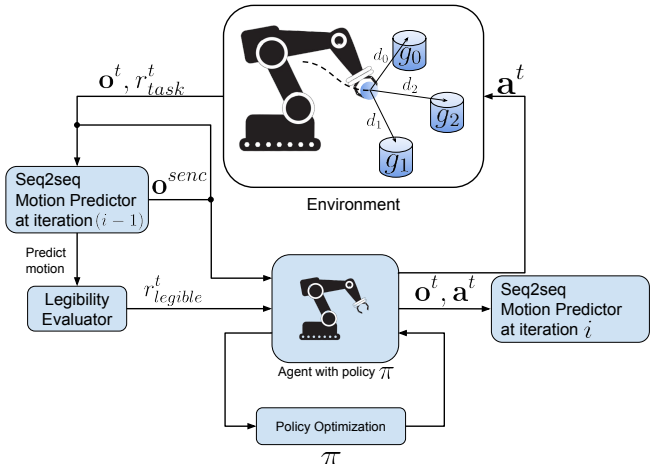

Fig. 2: An overview of our approach. At each time-step t, the agent received observation $\mathbf{o}^t = [\mathbf{o}^t_{task}, \mathbf{o}^t_{senc}]$ from environment and legible interpreter, reward $r^t = [r^t_{task}, r^t_{legibile}]$, and generates an action a^t following the policy π. The policy π is updated by a policy gradient based reinforcement learning algorithm. In addition, the data generated by agent with policy π at iteration i is sent to update the Seq2seq intent interpreter in iteration $i+1$.

1) Observation space: The observation vector at time-step t can be divided into 4 parts: $\mathbf{o}_t = [\mathbf{o}^\zeta_t, \mathbf{o}^g, \mathbf{o}^\mathbf{d}, \mathbf{o}^{senc}]$, where $\mathbf{o}^\zeta_t$ represents robot joint angle and end-effector trajectory at time-step t and $\mathbf{o}^g$ stands for its goal position. $\mathbf{o}^\mathbf{d}$ denotes the distances between end-effector to all goals in the goal set. In addition, $\mathbf{o}^{senc}$ represents the latent state space from seq2seq encoder. The observations are normalized by subtracting the mean and divided by the standard deviation using the statistics aggregated over the course of the entire training.

2) Action space: The action space is a set of permissible velocities in continuous space. The action of robot includes joint velocity of each joint. In this work, considering the real robot's kinematics and the real-world applications, we set the range of the joint velocity $v \in (-0.5, 0.5)\mathrm{rad/s}$ and the acceleration $acc \in (-0.1, 0.1)\mathrm{rad/s^2}$.

3) Reward design: Our objective is to successfully reach the manipulation goal as well as maximize the legibility of the trajectory. A reward function is designed to guide the robot to achieve this objective:

$$r^t = r^t_{task} + \omega_r \cdot r^t_{legible}.$$

The reward r at time-step t is a sum of two terms, task reward r^t_{task} and legible reward $r^t_{legible}$. ω_r is a weight coefficient for balancing between task reward and legible reward. In particular, the task reward consists of two components:

$$r^t_{task} = (g_r)^t + (c_r)^t.$$

The robot is awarded by $(g_r)^t$ for reaching its goal:

$$(g_r)^t = \begin{cases} r_{arrival}, & \text{if } \left\| \zeta_t^E - p_{g_0} \right\|_2 < 0.05; \\ 0, & \text{otherwise.} \end{cases}$$

When the robot collides with itself or obstacles in the environment, it is penalized by $(c_r)^t$:

$$(c_r)^t = \begin{cases} r_{collision}, & \text{if robot in collision;} \\ 0, & \text{otherwise.} \end{cases}$$

We set $r_{arrival} = 300.0$ and $r_{collision} = -30.0$ in the training procedure. The legible reward $r^t_{legible}$ is generated from one of the three legible functions which are described in Section III-B.

D. Motion Predictor

As mentioned, we use a Seq2Seq network for robot motion prediction. To reduce the network size, here we choose Gated Recurrent Unit (GRU) instead of LSTM. This network takes the end-effector position at last 10 time-steps as input and outputs the predict motion at next 20 time steps. All the data is normalized by Z-Score method. We use MSE as the loss function, which is formulated as:

$$\begin{aligned} L_{pred} &= L(o^{\boldsymbol{\zeta^E_{t-m:t}}}, f(o^{\zeta^E_{t-m-n:t-m}})) \\ &= MSE(o^{\zeta^E_{t-m:t}}) \end{aligned}$$

where

$$MSE(o^{\zeta^E_{t_0:t}}) = \frac{1}{t-t_0} \sum_{\tau=t_0}^{\tau=t} \left\| o^{\zeta^E_\tau} - \widehat{o}^{\zeta^E_\tau} \right\|_2$$

1) Network Architecture and Training process: Figure 3 shows the network architecture of the motion predictor and the policy network of motion planner. A Seq2Seq network is used for motion prediction. Both encoder and decoder have 2 layers, within each layer containing 8 GRU units. The policy network maps the latent space of encoder and environment observation to robot actions through two fully connected layers. The policy network is optimized using Proximal Policy Optimization [14].

The joint training procedure is shown in Algorithm 1. To initialize the predictor network, we first train a policy network with only fast approaching reward at 1_{st} iteration.

Then we start the joint training from line 12 in Algorithm 1. The policy network training is introduced in Algorithm 2. The hyperparameters of our training algorithm described in Algorithm 2 are: $lr_\theta = 1e-3$, $\gamma = 0.99$.

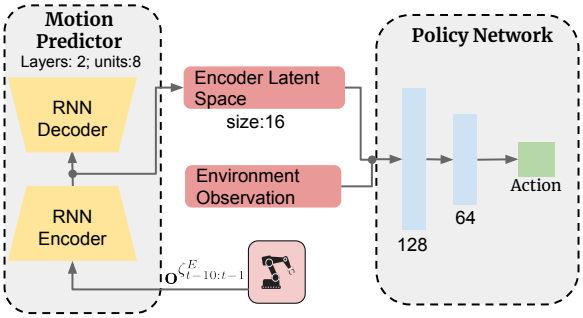

Fig. 3: The architecture of the legible motion planning network. The input of the moiton predictor is the robot end-effector positions in the last 10 time-steps. The policy network has the environment observation and encoder state $\mathbf{o} = [\mathbf{o}^{\zeta}_t, \mathbf{o}^{g}, \mathbf{o}^{\mathbf{d}}, \mathbf{o}^{senc}]$ as inputs, and outputs the mean of joint velocity $\mathbf{a}$ as the action command.

Algorithm 1 Actor-Critic Reinforcement Learning Framework for Motion Planning

Initialize a policy network π_θ, a value function V_ϕ, a predictor network P_ψ, and the trajectories buffer $\mathbb{B} = \emptyset$.
for $iteration = 1, 2, \cdots$, **do**
  // Policy Network Training;
  for $steps_{rl} = 1, 2, \cdots$, **do**
    for actor $i = 1, 2, \cdots, N$ **do**
      Run policy π_θ in environment for T_i time steps
      Collect trajectories $\tau_i = \{\mathbf{o}_i^{0:T_i}, \mathbf{a}_i^{0:T_i}, r_i^{0:T_i}\}$
    end for
    Train π_θ, V_{phi} with $\tau_{i:N}$ by alg. 2
    Add $\tau_{i:N}$ into the buffer $\mathbf{B}$
  end for
  // Predictor Network Training;
  for $steps_{pred} = 1, 2, \cdots$, **do**
    Sample minibatch of m sequences from buffer $\mathbb{B}$.
    Update the ψ with lr_ψ w.r.t. L_{pred}
  end for
  Clear the buffer $\mathbb{B} = \emptyset$
end for

IV. Experimental Evaluation

To evaluate our approach, we develop a human-robot interaction task in simulated environments and conduct the experiments with human participants. In this section, we first introduce the task design and show the experiment set-up. Then we introduce our objective and subjective measurements for human participants.

A. Task design

Since the human's prediction of the robot's future motion is difficult to be collected directly, we design a manipulation task to measure it indirectly, i.e., by using human's performance in terms of accuracy and waiting-time when estimating the manipulation task's goal region as side-channel signals for human's prediction. This is based on the fact that a better prediction of the robot motion would lead to a more accurate estimate of the goal region of the manipulation task.

Algorithm 2 Policy Network Training By Proximal Policy Optimization

Input: the policy network π_θ, the value function V_ϕ and a set of trajectories $\mathbb{D} = \tau_{1:N}$
Construct advantage estimates $\hat{A}$ [15] based on V_ϕ
$\pi_{old} \leftarrow \pi_\theta$
Update the θ with lr_θ by maximizing PPO objective with $\mathbb{D}$:
$\arg\max \frac{1}{|\mathbb{D}|} \sum \min(\frac{\pi_\theta(a|o)}{\pi_{old}(a|o)}\hat{A}, clip(\frac{\pi_\theta(a|o)}{\pi_{old}(a|o)}, 1-\epsilon, 1+\epsilon)\hat{A})$
Update the ϕ with lr_ϕ by regression on mean-squared error:
$\arg\min \frac{1}{|\mathbb{D}|} \sum (V_\phi(o) - \sum_{t'=t}^{T} \gamma^{t'-t} r^t)^2$

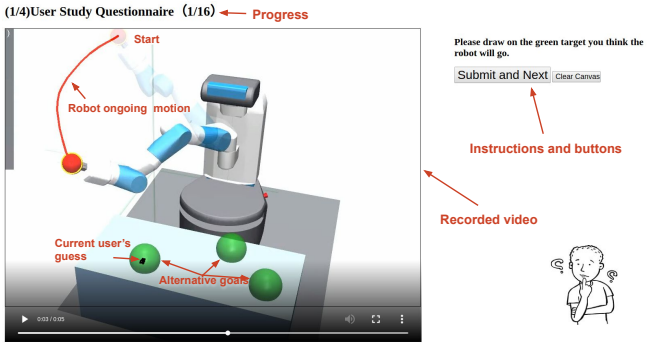

Fig. 4: Experiment setup and user interface. Red bold characters indicates the our added caption. The users watch the recorded robot video, while guessing the target at the same time. Participants are instructed to guess the target as early as possible and draw their guessing on the video.

In this task, the participants are informed that robots and humans are sitting on both sides of a table so the tabletop is a shared work space. The robot task is to put a red tool from a random position into the green balls as shown in Figure 4. The human is required to predict which green points the robot wants to approach. We described the task scenario to the participants as the robot is assembling green balls with a red tool, while the human needs to determine the goal green balls as soon as possible. This task is implemented in a simulated environment based on OpenAI Gym environment [16] and the physics engine is MuJoCo [17].

To evaluate human's prediction, we design a test interface that can play recorded robot motion videos and allow participants to draw their predictions as shown in Figure 4. The participants are instructed to use their best estimate of the robot's target and draw on the video. This test allows us to evaluate how well and fast the robot communicates its intent by measuring how fast and accurately the participants

estimate the robot's target.

The robot's starting configuration and target are randomly sampled from the valid configuration space and target space respectively. To ensure consistency of measurement across participants and to avoid potential bias in sampling toward one side of the workspace and target sets, we divide the workspace into the left and right parts according to the position of target in the middle. In particular, we divide the start-goal area combination into three categories based on the estimating difficulty, examples shown in Figure 5:

1) Same side of start-goal: Left area to left target; Right area to right target.
2) Different side of start-goal: Left area to right target; Right area to left target.
3) Middle target: Any start area to middle target.

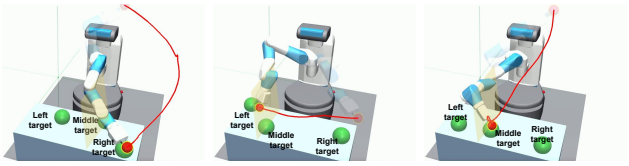

Fig. 5: Examples of start-goal area combination. From left to right: same side of start-goal; different side of start-goal; middle target.

During the recording, we set a random seed to make the sampling goal in a reproducible way. To avoid over difficult scenarios, we set the lower distance limitation between each target to $0.2\,\mathrm{m}$. We selected the first 30 recordings of each category to build a test set. To ensure consistency of measurement across participants, each participant saw randomly selected $3\times5=15$ movements from three categories of each method. Before the experiment, each participant is asked to read a description of the experiment and sign a consent form.

B. Training result

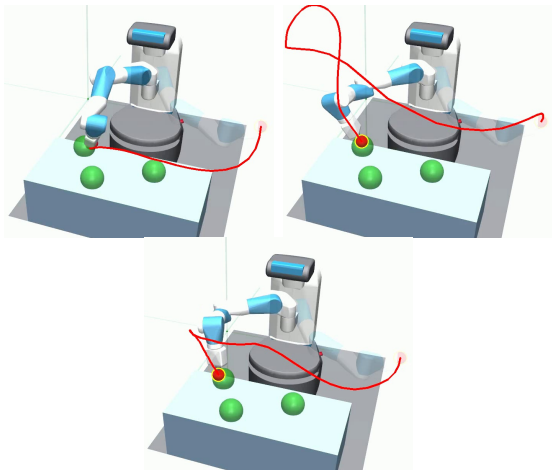

Fig. 6: Trajectory examples in same scenario for three methods. Upper-left: *FastApp*. Upper-right: *EffDist*. Bottom: *TrajDist*.

We train our network in the above scenario with three functions: *FastApp* ($\omega_r = 20$), *EffDist* ($\omega_r = 30$) and *TrajDist* ($\omega_r = 30$). The predictor validation loss, average task rewards and legible reward in training stage are shown in Figure 7. The reward of *FastApp* is highly relevant to task rewards and it has a different scale from others, so we do not show it. The middle figure shows that all four models can complete the task with about 300 train steps. The left figure indicates that the minimum validation loss of predictor decreases when the iteration number increases. The right figure shows that the policy network successfully optimizes the legible reward. Both left and right figure support that the predictor and the planner successfully learn to adapt to each other. Figure 6 shows trajectory examples in the same scenario for three methods. (For more examples please refer to: `https://sites.google.com/view/actor-critic-legible-planner`).

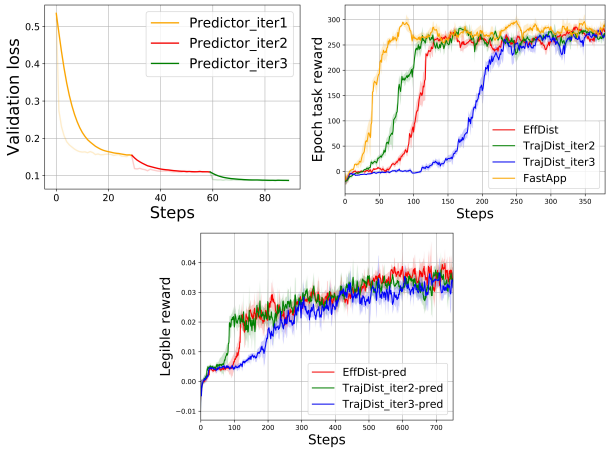

Fig. 7: Upper-left: validation loss of predictor at the training stage in different iterations. Upper-right: average task rewards shown in training steps during the training process. For each iteration we train PPO for around 300 steps. Bottom: average legible rewards shown in training steps during the training process.

C. Measures

We utilize objective and subjective measures that capture user performance, particularly how quickly and accurately participants infer the robot's goal from its motion, as well as the perceptions about the robot's motions, including how legible, learnable and natural they perceive the motions to be. These measures are further detailed below.

1) Objective measures: To evaluate the objective performance of both humans and Seq2seq models, we suggest three measures, in particular:

- **Trajectory ratio:** The perception of the trajectory **remaining** for the robot to complete the task when the participant is able to choose the correct target and does not change their guess again. Compared to Time ratio, this measure works better if the robot does not move at a uniform speed.
- **Time ratio:** The percentage of the **time duration** when the participant is able to choose the correct target and does not change their guessing again. time ratio $=$ time correct/duration.
- **User idle time:** The time between the start of the motion and the time the participant begins to choose the target.

2) Subjective measures: We divide the subjective measurement into 3 categories as shown in Table I. Each category includes 2 questions which are recorded with a five-point rating scale.

TABLE I: Subjective measures for interactive task performance.

Legibility:
(1) The robot motion clearly conveys its target to me.
(2) It was easy to predict which goal the robot wants to reach.
Learnability:
(3) I can learn to well predict the robot's motion after several trials.
(4) It was easier to guess the target of robot after several tests.
Naturalness:
(5) The robot's motion looked natural to me.
(6) The robot moved in a human-like way.

V. Results

We recruited 15 participants from the university campus for this study. The study involved 5 females and 10 males. Participants reported moderate familiarity ($M = 3.12, SD = 1.07$) with robots on a five-point rating scale. 4 Participants reported participating in prior robotics research studies. Participants were paid $40 HKD for approximately 20 minutes of participation. Eventually, we collected 197, 168 and 208 effective responses for *FastApp*, *EffDist* and *TrajDist* motion types, respectively.

A. Comparing robot motion versus functionality

From the collected data, none of the measures for the data we collected met the normality assumptions for analysis of variance after testing the goodness-of-fit of a normal distribution using the Shapiro-Wilk W test ($p < 0.001$ for all measures). Therefore, we analyzed our dataset using the Wilcoxon rank-sum test. We suggest 2 hypotheses:

- *H1. Perceptions of legibility* Robot motion type will *significantly* affect perceived legibility. Legibility ranking would be rated higher for both *EffDist* and *TrajDist* than *FastApp*.
- *H2. Perceptions of learnability* Robot motion type will *significantly* affect perceived Learnability. Learnability ranking would be rated highest for *TrajDist* followed by *EffDist* then *FastApp*.

TABLE II: Descriptive statistics.

Objective Measures	FastApp			EffDist			TrajDist		
	M	SD	Mdn	M	SD	Mdn	M	SD	Mdn
Trajectory ratio	0.149	0.197	0.058	0.259	0.199	0.209	0.297	0.223	0.258
Time ratio	0.724	0.247	0.796	0.526	0.213	0.509	0.517	0.244	0.494
User idle time	2.54	1.47	2.17	2.62	1.36	2.30	2.89	1.81	2.33
Subjective Measures	**FastApp**			**EffDist**			**TrajDist**		
	M	SD	Mdn	M	SD	Mdn	M	SD	Mdn
Legibility	2.90	0.94	3.00	3.80	0.91	4.00	4.00	0.58	4.00
Learnability	3.03	1.02	3.00	3.80	0.95	4.00	4.00	0.86	4.00
Naturalness	2.77	1.09	3.00	3.23	0.80	3.00	3.33	0.91	3.00

B. Comparison result

Table II provides the descriptive statistics of both objective and subjective measurements. Color depicts the ordering of results from dark blue (best performing) to light blue (worst performing). Table III shows pairwise comparisons for significant effects using the Wilcoxon rank-sum test. In general, our analysis shows that for all measures, *TrajDist* is either the best performing motion type or not significantly worse than the best motion type.

TABLE III: Pairwise comparisons for significant effects (p-value) of motion types.

Measure	FastApp *vs.* EffDist	TrajDist *vs.* FastApp	EffDist *vs.* TrajDist
Trajectory ratio	$< .001$	$< .001$	.131
Time ratio	$< .001$	$< .001$	.668
User idle time	.149	.056	.700
Legibility	$< .001$	$< .001$	.562
Learnability	.003	$< .001$	.373
Naturalness	.117	.056	.626

The results of objective measures indicate that our approach significantly improves on helping people better predict the robot's intent, especially the *TrajDist* method. Though the value of user idle time for *FastApp* motion is smallest, the small time ratio indicates that the users predicted the correct goal very late so their initial guesses were often wrong. For *FastApp* motion types, 4 users reported that they felt like the robot was purposely misleading them. For *EffDist* and *TrajDist*, the result of *time ratio* measure is similar, but the *Trajectory ratio* measure shows a much better result of *TrajDist* motion. This is caused by the non-uniform moving velocity along a trajectory. Besides, some participants reported that they felt the *EffDist* motions went higher and longer than others in special cases. According to users' feedback, we analyzed the ratio between trajectory length and start-goal distance based on 8000 random generated motions for each method. The result of this ratio is: *FastApp*: $M = 2.56, SD = 2.40$; *EffDist* $M = 3.60, SD = 2.67$ and *TrajDist* $M = 3.26, SD = 2.13$. Thus, *TrajDist* is 27% longer and *EffDist* is 40% longer than *FastApp*.

Our hypothesis regarding participants perceptions of legibility (H1) is supported by out results. Both *EffDist* and *TrajDist* are rated higher than *FastApp*. The results also show partial support for H2 that *TrajDist* is rated highest followed by *EffDist* and then *FastApp*.

VI. Conclusion

In this work, we suggest an actor-critic approach for legible robot motion planner and a legibility evaluator for 3D manipulation task. Results show that our approach significantly helps people better predict the robot's intent than "functional" motion. Without the usage of real-human data, the motion predictor trained from joint learning successfully help the planner generate more legible motion. In addition, considering prediction helps the robot avoid overly redundant trajectories.

There are several directions that we plan to explore in the future work. First, we will develop an direct way to accurately and promptly measure human's prediction of the robot's motion, e.g., by using an eye-tracking device. Second, we will enhance the current robot motion predictor with rewards reflecting task goal constraints in order to minimize the task-specific prediction error. Finally, we plan to recruit more participants to provide a more convincing user-study.

REFERENCES

[1] A. Edmondson, "Psychological safety and learning behavior in work teams," *Administrative science quarterly*, vol. 44, no. 2, pp. 350–383, 1999.

[2] S.-J. Blakemore and J. Decety, "From the perception of action to the understanding of intention," *Nature reviews neuroscience*, vol. 2, no. 8, p. 561, 2001.

[3] D. Bortot, M. Born, and K. Bengler, "Directly or on detours?: how should industrial robots approximate humans?" in *Proceedings of the 8th ACM/IEEE international conference on Human-robot interaction*. IEEE Press, 2013, pp. 89–90.

[4] A. D. Dragan, K. C. Lee, and S. S. Srinivasa, "Legibility and predictability of robot motion," in *Proceedings of the 8th ACM/IEEE international conference on Human-robot interaction*. IEEE Press, 2013, pp. 301–308.

[5] L. Takayama, D. Dooley, and W. Ju, "Expressing thought: improving robot readability with animation principles," in *2011 6th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. IEEE, 2011, pp. 69–76.

[6] M. J. Gielniak and A. L. Thomaz, "Generating anticipation in robot motion," in *2011 RO-MAN*. IEEE, 2011, pp. 449–454.

[7] S. Gu, E. Holly, T. Lillicrap, and S. Levine, "Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates," in *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2017, pp. 3389–3396.

[8] L. Pinto and A. Gupta, "Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours," in *2016 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2016, pp. 3406–3413.

[9] S. Lefèvre, D. Vasquez, and C. Laugier, "A survey on motion prediction and risk assessment for intelligent vehicles," *ROBOMECH journal*, vol. 1, no. 1, p. 1, 2014.

[10] C. Bodden, D. Rakita, B. Mutlu, and M. Gleicher, "A flexible optimization-based method for synthesizing intent-expressive robot arm motion," *The International Journal of Robotics Research*, vol. 37, no. 11, pp. 1376–1394, 2018.

[11] R. M. Holladay, A. D. Dragan, and S. S. Srinivasa, "Legible robot pointing," in *The 23rd IEEE International Symposium on Robot and Human Interactive Communication*. IEEE, 2014, pp. 217–223.

[12] M. Zhao, R. Shome, I. Yochelson, K. Bekris, and E. Kowler, "An experimental study for identifying features of legible manipulator paths," in *Experimental Robotics*. Springer, 2016, pp. 639–653.

[13] B. Busch, J. Grizou, M. Lopes, and F. Stulp, "Learning legible motion from human–robot interactions," *International Journal of Social Robotics*, vol. 9, no. 5, pp. 765–779, 2017.

[14] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[15] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," *arXiv preprint arXiv:1506.02438*, 2015.

[16] M. Plappert, M. Andrychowicz, A. Ray, B. McGrew, B. Baker, G. Powell, J. Schneider, J. Tobin, M. Chociej, P. Welinder, V. Kumar, and W. Zaremba, "Multi-goal reinforcement learning: Challenging robotics environments and request for research," 2018.

[17] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 5026–5033.