2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 1-5, 2023. Detroit, USA

Learning Joint Policies for Human-Robot Dialog and Co-Navigation

Yohei Hayamizu[1], Zhou Yu[2], and Shiqi Zhang[1]

***Abstract*— Service robots need language capabilities for communicating with people, and navigation skills for beyond-proximity interaction in the real world. When the robot explores the real world with people side by side, there is the compound problem of human-robot dialog and co-navigation. The human-robot team uses dialog to decide where to go, and their shared spatial awareness affects the dialog state. In this paper, we develop a framework that learns a joint policy for human-robot dialog and co-navigation toward efficiently and accurately completing tour guide and information delivery tasks. We show that our approach outperforms baselines from the literature in task completion rate and execution time, and demonstrate our approach in the real world.**

I. Introduction

Robots are increasingly tasked with services that require dialog and navigation in human-inhabited environments [1]. While there is rich literature on mobile robot navigation [2] and dialog systems [3], there is relatively little research on the joint management of human-robot dialog and their co-navigation. Considering a robot that moves around while conversing with a human side by side, e.g., a robot realtor, the robot needs the capability of "dialog navigation" [4], [5]. The dialog navigation task requires the robot to navigate and interact with a human at the same time, while fulfilling service requests.

When a robot talks to and navigates with people at the same time, there are new challenges caused by the interplay between the two types of actions. For instance, when a real estate robot wants to tell a customer that a bedroom is large and cozy, which requires human-robot dialog, it is better for the robot to first guide the customer to that bedroom, which requires human-robot co-navigation. Dialog capabilities enable the robot to estimate the human's belief state, and accordingly select the next navigation goal; navigation capabilities help change the human-robot system's location to facilitate the next few turns of the ongoing dialog. Many studies on dialog navigation allow remote communication between a user and a robot, e.g., the user is from a call center [6], [7], where the complexity of social navigation is avoided. By comparison, we consider scenarios where the human and robot talk to and navigate with each other at the same time.

In this paper, we focus on dialog navigation tasks where the human and robot are not always co-located. In such tasks, tracking the positional information and dialog states is crucial for the robot. For instance, the robot might want to navigate to the human, and then verbally encourage the human to walk together to a room before discussing the room features. We develop a mobile robot system that leverages the locations of the human and itself for selecting its dialogue and navigation behaviors. In order to deal with the uncertainty and partial observability in dialog and navigation actions, we use Deep Recurrent Reinforcement Learning for computing joint policies for selecting both dialog and navigation actions [8], [9]. Our framework enables the robot to reason about the history of observations from dialog and navigation actions toward the long-term goal of efficiently and accurately conveying a few location-dependent statements.

Fig. 1: When the robot receives positive feedback from the human, the robot believes it is likely that the information (the room being `beautiful` and with `view`) has been successfully delivered (**top-left**). In the absence of the human, it is unlikely the robot is able to deliver the room features (**top-right**). When the human does not give any verbal feedback to the robot, the likelihood that the robot successfully delivers the information is medium (**bottom-left**). When the robot talks about another room instead of the current one, there is a lower chance that the human accepts such information (**bottom-right**).

We have evaluated our framework in a real estate agent domain where a mobile robot is tasked with fulfilling user requirements of house features via dialog and navigation actions. Our system has been compared with three baselines that alternatively take dialog and navigation actions. One has a rule-based dialog-navigation planner, and another relies on a dialog policy and a planner for navigation. Our experiments show that our framework can learn a joint policy and outperform the baselines regarding task completion rate and execution time.

[1] SUNY Binghamton
Email: {yhayami1; zhangs}@binghamton.edu
[2] Columbia University
Email: zy2461@columbia.edu

2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) | 978-1-6654-9190-7/23/$31.00 ©2023 IEEE | DOI: 10.1109/IROS55552.2023.10341663

II. Related Work

This section discusses three research areas that are related to dialog navigation systems: (1) instruction-following navigation systems, (2) systems where navigation follows dialog, and (3) systems where navigation and dialog are interleaved.

Instruction Following: Instruction-following navigation systems have been studied, where a robot communicates with a user to complete a navigation task based on user instructions. Kollar et al. developed a system that can generate action plans from a given directional instruction by parsing the natural language [10]. Other works introduced parser learning for indoor navigation instruction that translates natural language commands to actions [11], [12]. In recent years, researchers developed a benchmark, ALFRED, for learning to ground both language instructions and robot perceptions into sequences of actions [13], realizing complex robot tasks such as household tasks [14], [15]. Those works assumed the human and robot share the same observability over the world and focused on understanding human instructions instead of managing multi-turn conversations.

Navigation Goal Specification via Dialog: One way of integrating dialog and navigation is to specify navigation goals via human-robot dialog, where the dialog occurs before navigation. Some efforts enabled a robot to conduct a service task after identifying user requests through dialog with probabilistic inference [16], [17], [18]. Other works learned dialog behaviors, such as asking clarification questions from human-robot conversations, to improve the task performance [5], [19]. Another work focused on spoken language understanding to retrieve navigation-associated information and improve a navigation task that follows the language understanding step [20]. Although such systems improved the performance in task completion by introducing a dialog system into navigation tasks, the robot cannot handle dynamic changes in user requests since the conversation happens only before navigation in their systems. In addition, those works assume that conversations happen in a single location, whereas the multiple turns of our human-robot dialog might occur in different locations.

Interleaved Dialog Navigation: Recent research has produced dialog navigation systems that alternate between dialog and navigation actions. Thomason et al. presented Cooperative Vision-and-Dialog Navigation (CVDN) that trains language-teleoperated home and office robots that ask targeted questions about where to go next [21]. DIALFRED [22] is an extension of ALFRED that learns to ask for instructions to handle unexpected situations. Studies on such dialog navigation systems have developed some datasets to study navigation and spatial reasoning with real-life observations, learning to ground language to perception and behavior, and realizing a cooperative localization task [23], [7], [24], [6]. These studies can be seen as interactive instruction-following systems and allow instructions always to be available no matter where two agents are. By comparison, our robot can co-navigate with a human to fulfill service requests. Our work considers the robot as a guide and the human as a customer to study how the robot acquires complementary behaviors of dialog and navigation, being aware of the human's locations.

III. Deep Recurrent Reinforcement Learning

This section briefly describes deep recurrent RL. RL is an algorithm or a problem based on Markov Decision Process (MDP). An MDP is a tuple of $\langle S, A, T, R, \gamma \rangle$, where S is a set of states; A is a set of actions; $T(s, a, s') \in [0, 1]$ is a transition probability; $R(s, a) \in \mathbb{R}$ is an immediate reward; and $\gamma \in [0, 1)$ is the discount factor. RL algorithms aim to learn an optimal policy $\pi^* : S \rightarrow A$ by maximizing the expected cumulative return [25]. A policy is calculated by the Q-value function $Q(s, a)$ that estimates the expected cumulative return. While MDPs assume full observability of the states, the Partially Observable MDP (POMDP) can handle the uncertainty in the state space. A POMDP is a tuple of $\langle S, A, O, T, R, \gamma, Z \rangle$, where O is a set of observations, and $Z(s, a, o) \in [0, 1]$ is an observation probability. The objective of a POMDP problem is to find $\pi^* : B \rightarrow A$. B forms a belief space where a belief state $b(s)$ is calculated with the probability distribution of all possible states [26].

The deep Q-Network (DQN) developed by Mnih et al. [27] uses a deep neural network (DNN) to approximate the Q-value function. To learn the Q-value function $Q(s, a; \theta)$, DQN samples K experience tuples of (s, a, r, s') from the experience replay buffer memory and updates θ with the gradient of the following loss function $L(\theta)$.

$$L(\theta) = ((r + \gamma max_{a'} Q(s', a'; \theta^-)) - Q(s, a; \theta))^2,$$

where θ is a behavior network used for the action selection, and θ^- is a target network treated as the ground truth for the behavior network. Proximal Policy Optimization (PPO) is a method that can directly learn an optimal policy by estimating a policy gradient using DNN [28]. After sampling K experience tuples through trial-and-error experiences, PPO updates θ with the gradient. A general form of the loss function for PPO is as follows.

$$L(\theta) = min(\nabla(\theta)A, clip(\nabla(\theta), 1 - \epsilon, 1 + \epsilon)A),$$

where $\nabla = \pi(a; s, \theta))/\pi(a; s, \theta_{old})$ is a ratio of the probability under the new policy π and old policy π_{old}; A is the advantage function that estimates the current policy in the same manner as the Q-value function above; and ϵ is a hyperparameter of the clipping function which restrict drastic change in a policy. Those algorithms achieve human-level performance in many domains.

Deep Recurrent Reinforcement Learning algorithms are extensions of deep RL, introducing a Long Short-Term Memory (LSTM) [29] layer at the output layer. The LSTM can be regarded as an approximator of the belief state that can track historical information from observations to solve POMDP problems. This architecture was proposed by Hausknecht et al. [8] and extended to various derivatives. Pleines et al. described how a recurrent neural network could be incorporated into deep RL by applying an LSTM layer to

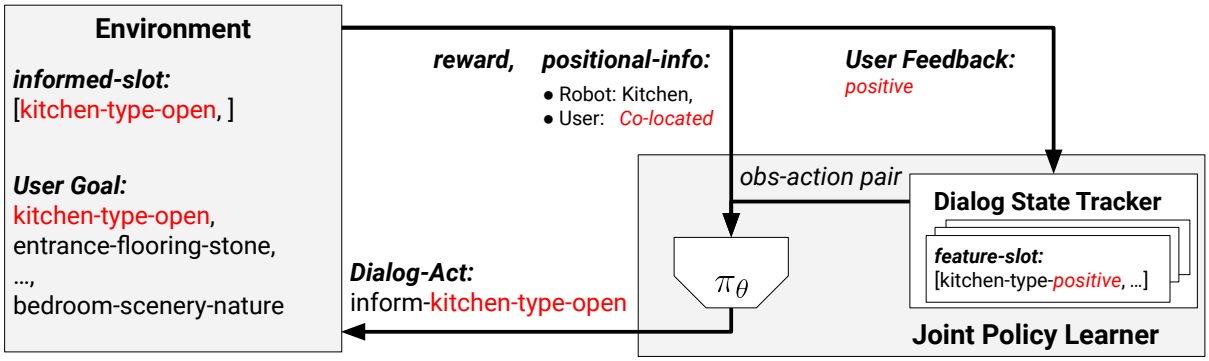


Fig. 2: An overview of our joint policy learning framework.

PPO (RecurrentPPO) and examined the limitations of deep recurrent reinforcement learning algorithms [9].

IV. Problem Statement

We design a real estate agent domain for a mobile dialog task where a robot guides users physically and performs dialog at the same time. Fig. 1 shows how a user and a robot interact with each other in our proposed domain. We define a domain description and problem for the mobile dialog task.

A. Real Estate Agent Domain

A domain description is defined as a tuple $\langle \mathbf{E}, \mathbf{e}, \mathbf{Q}, \mathbf{q} \rangle$, where $\mathbf{E} = [E_0, E_1, \cdots]$ is a finite set of entities of which a system can inform. For instance, $E_i \in \mathbf{E}$ can be "*livingroom-view*." $\mathbf{e} = [e_0, e_1, \cdots]$ is a full assignment of $\mathbf{E}$, which specifies the feature values of a particular house. For instance, $e_i =$ "*beautiful*" means the living room of the house has a beautiful view. $\mathbf{Q} = [Q_0, Q_1, \cdots]$ is a set of entities in which a user is interested, and $\mathbf{Q} \subseteq \mathbf{E}$. For instance, $Q_j \in \mathbf{Q}$ can be "*bedroom-size*." $\mathbf{q}$ is a full assignment of $\mathbf{Q}$, which specifies the requirement values of a particular user. For instance, if the user looks for a large bedroom, $q_j =$ "*large*".

B. Real Estate Agent Problem

The goal in the real estate agent domain is to estimate the user's requirements and introduce property features within a maximum number of timesteps. We define the real estate agent problem as a POMDP. $S := N \times N \times C^{|\mathbf{E}|} \times C^{|\mathbf{Q}|}$ is defined as a set of states factored in a robot location, a user location, entities to be informed to a user, and the user's requirements, where N is the number of rooms in real estate. The state space includes an entity slot and a requirement slot of which sizes are $|\mathbf{E}|$ and $|\mathbf{Q}|$, each with cardinality C. $A := A^D \cup A^N$ is a union of dialog actions A^D and navigation actions A^N. The types of actions are "*guidance*," "*inform*," and "*report*" for A^D and "*goto*" for A^N. When taking an *inform* action, the robot assigns a slot-value of an entity to the dialog action to inform the user about the entity, e.g., "*inform-livingroom-view-beautiful*". When taking other types of actions, the robot assigns a room to the action for navigation, e.g., "*guidance-bedroom-none-none*" and "*goto-bedroom-none-none*." Thus, the total number of actions is proportional to the number of rooms and entities. O is a set of observations the robot receives from the environment with a user. The observation contains the robot's and the user's locations and feedback from the user. The user gives *positive* feedback if the informed value $e_i \in \mathbf{e}$ meets the user's requirements $q_i \in \mathbf{q}$, *negative* feedback if not, and *none* otherwise. $T := S \times A \times S \to [0, 1]$ is a transition function. At the time-step t, the environment is in some state s_t. The robot takes an action a_t, which causes the environment to transition to state s_{t+1} with probability $T(s_t, a_t, s_{t+1})$. $Z := A \times S \times O \to [0, 1]$ is a observation function. The robot receives an observation $o_t \in O$, which depends on s_{t+1} and a_t, with probability $Z(o_{t+1}|a_t, s_{t+1})$ at the time-step t. $R := S \times A \to \mathbb{R}$ is a reward function. When the robot terminates the task, the reward function returns a reward according to the evaluation function $y(e, q) = |\{e_i | e_i \in q_i, 0 \leq i < |\mathbf{q}|\}|$. The more requirements of the user are satisfied, the higher rewards the robot receives. The robot receives a negative reward at each time-step as a living cost.

V. Proposed Approach

In this section, we present the key contribution of this work, an RL-based framework for learning a joint dialog-navigation policy.

Framework Description: In a dialog navigation task where the conversation and the positional relationship between a robot and a user strongly relate to their locations, optimization of navigation and dialog actions becomes more complex. We develop a framework that can learn a joint policy of dialog and navigation considering the spatial context and the course of dialogues. Joint policy learning refers to a model that maps a state that includes spatial and dialogue historical information to either a navigation or a dialog action. Thus, we propose unifying the spatial information with dialog state tracking so that the tracker can handle location changes during the dialogues. This formulation enables a robot to traverse different locations that bring new spatial contexts while tracking a dialog objective. We aim to compute a joint policy that guides the robot in the dialog and navigation actions toward maximizing the expected cumulative utility to achieve a mobile dialog task.

TABLE I: Examples of slot-value pairs for all domains

Domain	$\|\mathbf{E}\|$	Examples of {slot}-{value} pairs
livingroom	4	{view}-{beautiful}, {size}-{200}, $\cdots$
kitchen	4	{view}-{open}, {roomrel}-{livingroom}, $\cdots$
bedroom	4	{size}-{100}, {roomrel}-{livingroom}, $\cdots$

TABLE II: Dialog acts and navigation actions

User's actions	posans / negans / none
User's navigation actions	stay / goto
Robot's dialog actions	inform / guidance / report
Robot's navigation actions	stay / goto

Fig. 2 shows an overview of our framework. The framework consists of a joint policy learner and an environment. The joint policy learner has a dialog state tracker that tracks a course of dialogue and a dialog navigation policy that takes action based on the positional information and the dialog history. The environment includes a house instance and a user. The robot can take a verbal or navigation action, and the environment gives the robot feedback from the user and rewards based on the user's requirements.

Dialog State Tracking: In order for a robot to follow the course of dialogue over a task, we use a dialog state tracker. We utilize the slot-filling approach to modify the input to the dialog navigation policy instead of using raw observations so that the robot can be aware of entities to be informed. This allows the robot to store the current task completion rate over domains. The slots are updated by the dialog state tracker that takes user feedback and performs slot-filling. The dialog state tracker fills a corresponding slot with feedback from the user based on the observation.

We then use a robot's and user's locations, the feature slots, and the previous action as an observation o_t and input them to the dialog navigation policy network π_θ. Since the user's location is partially observable, the policy network needs to have a memory for the course of navigation. It is also important for the joint policy to deal with the user's feedback fluctuation due to uncertainty as a natural language understanding module propagates the feedback error to the dialog state tracking. According to the aforementioned reasons, we implement a policy network with a Deep Recurrent Reinforcement Learning model.

Implementation: Given the observation o_t, r_t, the dialog navigation policy network learns a joint policy that can output either a dialog or navigation action. Since navigation actions do not induce user feedback, the model requires a long memory for tracking an entire course of dialogue. Therefore, the policy network has an LSTM layer to aggregate observations over the turns. The policy network π_θ is updated every M time-step after collecting experiences from the environment. This process varies depending on the algorithm of choice. RecurrentPPO is an extension of PPO and has LSTM at the output layer [9]. RecurrentPPO collects M samples in its rollout buffer and calculates a policy gradient using the samples to update a network that selects the next action, an actor-network.

VI. Experiment

This section presents experimental results in simulation and a demonstration of our system in the real world. We develop a simulator for the evaluation of algorithms for dialog navigation tasks. We have compared our approach to three baselines and analyze the results.

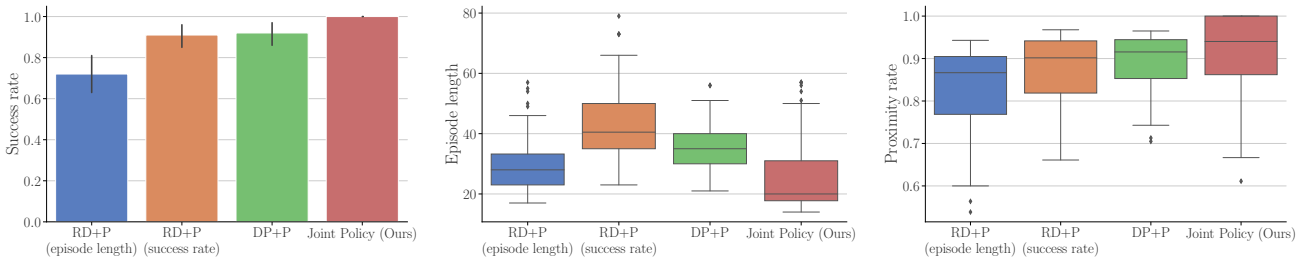
A. Simulator Design

We develop a simulated environment for the real estate agent problem. The simulator consists of 10 house instances. Each house instance has three domains ("*livingroom*", "*kitchen*", and "*bedroom*"), each with 4 feature slots. We manually created the slot-value pairs for each house instance. TABLE I shows examples of entities and those values for the three domains. Slot-value pairs are randomly assigned to the user's requirement slots when an environment is instantiated. At every time step, the robot can only receive its location, the user's existence, and the user feedback if available. Note that the user's location and the feedback from the user are available only when the robot and the user are co-located.

User simulators have been widely applied to training dialog policies using RL methods. In line with previous research, we simulate user behaviors, including responding to the robot's dialog actions and moving in the environment. During a mobile dialog task, the robot and the user act alternatively. TABLE II shows the robot and user actions used in the experiments. We use an action representation that is similar to the MultiWOZ-2.1 dataset for both dialog and navigation actions [30]. We consider different locations as domains as we consider an embodied system and address changes derived from the spatial context. The action representation contains 4 different pieces of information. The first piece is an *intent* which can be treated as an action. The second is a *domain*. Each domain has entities with values, and actions should include a *slot* and a *value* for an entity. For instance, the robot action, [*inform-kitchen-view-open*], represents that the robot *inform*s of a *kitchen view*, which is an *open* view. This action representation enables the joint policy learner to take both a dialog and navigation action as a part of observation in the same manner and simplifies the observation space representation and action space representation. An example of a course of dialog and location transitions is as follows:

An example of our human-robot dialog:

Robot: [*inform-livingroom-size-large*]
User: [*positive-livingroom-size-large*]
Robot: [*guidance-kitchen-none-none*]
User: [*stay-kitchen-none-none*]
Robot: [*goto-kitchen-none-none*]
User: [*goto-kitchen-none-none*]
$\cdots$
User: [*positive-bedroom-roomrel-kitchen*]
Robot: [*report-none-none-none*]



(a) Success rate. (b) Episode length. (c) Proximity rate

Fig. 3: Average success rate, the episode length, and proximity rate over 100 runs, while the robot works on completing the task for different users. The agent with the joint policy outperforms the baselines.

An example of trajectories of a robot and a user:

Robot: "*livingroom*" → "*kitchen*" → "*bedroom*"
User: "*livingroom*" → "*kitchen*" → "*bedroom*"

The environment returns a positive reward $R_{max} = 1.0$ when all user requirements are satisfied and a negative reward $R = -0.01$ otherwise. User feedback has noise with a 0.1 probability of being opposite. Robot and user movements are controlled with transition probabilities. The robot can successfully navigate to its goal location in 0.9 probability. The user follows the robot in presence of a 0.05 probability noise, i.e., there is a small probability that the user moves to a location that is different from the robot's goal location.

B. Experiment Settings

The robot spends $5,000,000$ timesteps for each run to learn a joint policy. The actor-network and the critic network have the same network architectures. The first two layers have an MLP layer of 64 units with *tanh* activation as a hidden layer. The output of the second layer is fed into the LSTM layer with 256 units. For hyper-parameters of RecurrentPPO, we vectorize the environment with the size of 32 to collect enough samples and use the samples with a batch size of 4096. The discounting factor γ is 0.99, the learning rate α is 0.002, and clip size ϵ is 0.1.

We compare the agent with a joint policy, *Joint Policy*, to three baselines: the agent with a dialog policy and navigation planner, *DP+P*, and the two agents with a rule-based dialog system and navigation planner. For DP+P, we use the same dialog state tracker and a RecurrentPPO model as our joint policy. The difference from our model is that the model takes only the dialog state and outputs a dialog action and does not consider the user location. We evaluate the agent's performance of those policies for 100 independent episodes every $100,000$ timestep by using a deterministic policy and select the best policy model to compare the performance in the success rate and the episode length. For the rule-based dialog systems, we designed two different strategies. One is to try to finish a task as soon as possible, *RD+P (episode length)*. The other is to try to achieve a higher success rate, *RD+P (success rate)*. Baseline agents have isolated dialog and navigation systems and conduct the task by executing dialog and navigation actions interleaved. The navigation planner of the baselines always takes the shortest plan from the current position to the destination.

C. Results

Fig. 3a and 3b show the results of the agent's success rate and episode length over 100 runs. Both figures demonstrate that our approach outperforms the baselines. Joint Policy could achieve the highest success rate and a shorter episode length. Although RD+P (episode length) has a lower episode length than RD+P (success rate) and DP+P, the success rate is the lowest among the four methods. DP+P produced a lower episode length than RD+P (success rate), whereas it has similar performance in the success rate. This observation shows that rule-based dialog systems need to spend a longer episode length to identify the user's requirements and that the dialog policy can track the dialog state properly The result also implies that DP+P could not achieve a higher success rate than RD+P (success rate) because of fewer co-located cases. In order to investigate how the robots handled both dialog and co-navigation, we define the proximity rate PR as follows:

$$PR = \frac{\textit{The number of turns when they are co-located}}{\textit{Total number of turns}}$$

The proximity rate presents how frequently the robot and the user are co-located. A higher proximity rate indicates that the robot can perceive the absence of the user and correctly select a navigation action so that they are co-located. Joint Policy has the highest proximity rate, and DP+P and RD+Ps have a lower proximity rate. This indicates that the DP+P tries to inform entities without considering the user's location to minimize the episode length, resulting in a lower success rate than Joint Policy. On the other hand, a higher proximity rate of Joint Policy indicates that our model could handle the positional information and dialog simultaneously, resulting in the highest success rate and the shortest episode length.

D. Demonstration in the Real World

We have demonstrated our approach using a real robot. We deployed our joint policy model on a segway-based mobile robot. The robot interacts with a human while traversing an environment where three rooms exist. We use YOLO V3 for ROS to realize real-time object detection [31] and

Google Dialogflow for a natural language understanding component [32].

VII. Conclusion and Future Work

In this paper, focusing on efficient and accurate human-robot dialog and co-navigation that consider the positional information and a course of dialogue, we develop a framework that learns a joint policy that maps an observation to either a dialog or navigation action. We evaluate our framework in a real estate agent domain. The goal is to guide a user while informing possible requirements of users that are not ever accessible. The experimental results show that our framework can learn a joint policy that outperforms the baselines in task completion rate and execution time. In the future, the low sample efficiency issue can be addressed to increase the cardinality of slots. Another direction is to enable end-to-end learning by further incorporating language understanding and perception into the loop.

ACKNOWLEDGMENTS

A portion of this work has taken place at the Autonomous Intelligent Robotics (AIR) Group, SUNY Binghamton. AIR research is supported in part by grants from the National Science Foundation (NRI-1925044), Ford Motor Company (URP Award 2019-2023), OPPO (Faculty Research Award 2020), and SUNY Research Foundation.

References

[1] S. Tellex, N. Gopalan, H. Kress-Gazit, and C. Matuszek, "Robots that use language," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 3, pp. 25–55, 2020.

[2] F. Gul, W. Rahiman, and S. S. Nazli Alhady, "A comprehensive study for robot navigation techniques," *Cogent Engineering*, vol. 6, no. 1, p. 1632046, 2019.

[3] S. J. Young, "Probabilistic methods in spoken–dialogue systems," *Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, vol. 358, no. 1769, pp. 1389–1402, 2000.

[4] H. Asoh, Y. Motomura, I. Hara, S. Akaho, S. Hayamizu, and T. Matsui, "Combining probabilistic map and dialog for robust life-long office navigation," in *1996 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, vol. 2, 1996, pp. 807–812 vol.2.

[5] J. Thomason, S. Zhang, R. J. Mooney, and P. Stone, "Learning to interpret natural language commands through human-robot dialog," in *24th International Joint Conference on Artificial Intelligence*, 2015.

[6] H. Chen, A. Suhr, D. Misra, N. Snavely, and Y. Artzi, "Touchdown: Natural language navigation and spatial reasoning in visual street environments," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 12 538–12 547.

[7] M. Hahn, J. Krantz, D. Batra, D. Parikh, J. M. Rehg, S. Lee, and P. Anderson, "Where are you? localization from embodied dialog," 2020.

[8] M. Hausknecht and P. Stone, "Deep recurrent q-learning for partially observable mdps," in *2015 aaai fall symposium series*, 2015.

[9] M. Pleines, M. Pallasch, F. Zimmer, and M. Preuss, "Generalization, mayhems and limits in recurrent proximal policy optimization," *arXiv preprint arXiv:2205.11104*, 2022.

[10] T. Kollar, S. Tellex, D. Roy, and N. Roy, "Toward understanding natural language directions," in *2010 5th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. IEEE, 2010, pp. 259–266.

[11] D. Chen and R. Mooney, "Learning to interpret natural language navigation instructions from observations," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 25, no. 1, 2011, pp. 859–865.

[12] C. Matuszek, E. Herbst, L. Zettlemoyer, and D. Fox, "Learning to parse natural language commands to a robot control system," in *Experimental robotics: the 13th international symposium on experimental robotics*. Springer, 2013, pp. 403–415.

[13] M. Shridhar, J. Thomason, D. Gordon, Y. Bisk, W. Han, R. Mottaghi, L. Zettlemoyer, and D. Fox, "Alfred: A benchmark for interpreting grounded instructions for everyday tasks," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 10 740–10 749.

[14] S. Y. Min, D. S. Chaplot, P. Ravikumar, Y. Bisk, and R. Salakhutdinov, "Film: Following instructions in language with modular methods," *arXiv preprint arXiv:2110.07342*, 2021.

[15] V.-Q. Nguyen, M. Suganuma, and T. Okatani, "Look wide and interpret twice: Improving performance on interactive instruction-following tasks," *arXiv preprint arXiv:2106.00596*, 2021.

[16] S. Zhang and P. Stone, "Corpp: Commonsense reasoning and probabilistic planning, as applied to dialog with a mobile robot," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 29, no. 1, 2015.

[17] D. Lu, S. Zhang, P. Stone, and X. Chen, "Leveraging commonsense reasoning and multimodal perception for robot spoken dialog systems," in *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2017, pp. 6582–6588.

[18] Y. Chen, F. Wu, W. Shuai, and X. Chen, "Robots serve humans in public places—kejia robot as a shopping assistant," *International Journal of Advanced Robotic Systems*, vol. 14, no. 3, p. 1729881417703569, 2017.

[19] S. Amiri, S. Bajracharya, C. Goktolgal, J. Thomason, and S. Zhang, "Augmenting knowledge through statistical, goal-oriented human-robot dialog," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 744–750.

[20] Y. Zheng, Y. Liu, and J. H. Hansen, "Navigation-orientated natural spoken language understanding for intelligent vehicle dialogue," in *2017 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2017, pp. 559–564.

[21] J. Thomason, M. Murray, M. Cakmak, and L. Zettlemoyer, "Vision-and-dialog navigation," in *Conference on Robot Learning*. PMLR, 2020, pp. 394–406.

[22] X. Gao, Q. Gao, R. Gong, K. Lin, G. Thattai, and G. S. Sukhatme, "Dialfred: Dialogue-enabled agents for embodied instruction following," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 10 049–10 056, 2022.

[23] A. Padmakumar, J. Thomason, A. Shrivastava, P. Lange, A. Narayan-Chen, S. Gella, R. Piramuthu, G. Tur, and D. Hakkani-Tur, "Teach: Task-driven embodied agents that chat," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 2, 2022, pp. 2017–2025.

[24] H. De Vries, K. Shuster, D. Batra, D. Parikh, J. Weston, and D. Kiela, "Talk the walk: Navigating new york city through grounded dialogue," *arXiv preprint arXiv:1807.03367*, 2018.

[25] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[26] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and acting in partially observable stochastic domains," *Artificial intelligence*, vol. 101, no. 1-2, pp. 99–134, 1998.

[27] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.

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

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

[30] M. Eric, R. Goel, S. Paul, A. Kumar, A. Sethi, P. Ku, A. K. Goyal, S. Agarwal, S. Gao, and D. Hakkani-Tur, "Multiwoz 2.1: A consolidated multi-domain dialogue dataset with state corrections and state tracking baselines," *arXiv preprint arXiv:1907.01669*, 2019.

[31] M. Bjelonic, "YOLO ROS: Real-time object detection for ROS," https://github.com/leggedrobotics/darknet_ros, 2016–2018.

[32] N. Sabharwal, A. Agrawal, N. Sabharwal, and A. Agrawal, "Introduction to google dialogflow," *Cognitive virtual assistants using google dialogflow: develop complex cognitive bots using the google dialogflow platform*, pp. 13–54, 2020.