General Dynamic Goal Recognition

Osher Elhadad[1] **and Reuth Mirsky**[1, 2]

[1]Bar Ilan University, Israel [2]Tufts University, MA, USA
osher.elhadad@live.biu.ac.il, reuth.mirsky@tufts.edu

Abstract

Understanding an agent's intent through its behavior is essential in human-robot interaction, interactive AI systems, and multi-agent collaborations. This task, known as Goal Recognition (GR), poses significant challenges in dynamic environments where goals are numerous and constantly evolving. Traditional GR methods, designed for a predefined set of goals, often struggle to adapt to these dynamic scenarios. To address this limitation, we introduce the General Dynamic GR problem – a broader definition of GR – aimed at enabling real-time GR systems and fostering further research in this area. Expanding on this foundation, this paper employs a model-free goal-conditioned RL approach to enable fast adaptation for GR across various changing tasks.

Introduction

Goal Recognition (GR) is a subarea of artificial intelligence (AI) focused on understanding and predicting the goals of agents based on their actions. This task is essential in various fields, particularly in human-robot interaction (Massardi, Gravel, and Beaudry 2020; Trick et al. 2019; Scassellati 2002) and multi-agent systems (Rabkina and Forbus 2019; Kaminka, Wendler, and Ronen 2001; Sukthankar and Sycara 2011; Bansal et al. 2019), as it plays a crucial role in understanding agent behaviors.

Most traditional GR solutions primarily address single GR tasks with a specific set of goals within a single environment. However, they must often restart the process when presented with a new set of goals within the same domain or an entirely new domain. This restart adds significant time overhead (e.g., reapplying planners or RL for each new goal), rendering these approaches impractical in real-time scenarios where the goal space is continuous or there is a need for rapid adaptation to dynamically changing GR tasks with diverse goals and domains. For example, in assistive technologies for the elderly or individuals with motor disabilities (Zhang et al. 2017), a robotic assistant might need to dynamically adjust to a range of objectives – such as inferring human needs for physical tasks like fetching items or responding to urgent medical requirements. Similarly, in autonomous vehicle systems (Brewitt et al. 2023), vehicles must continuously adapt and infer the objectives of surrounding traffic agents, such as pedestrians, other vehicles, and drivers' behaviors.

This paper addresses these limitations by introducing a **General Dynamic Goal Recognition (GDGR)** framework. It achieves this by (1) proposing a new definition of General Dynamic Goal Recognition, (2) outlining a general approach for GDGR, and (3) applying goal-conditioned, model-free Reinforcement Learning (RL) to enable real-time GR across multiple dynamically changing tasks within a single domain.

We present preliminary results from the Point Maze (Towers et al. 2024) environment (shown in Figure 1). The results indicate significant improvements in adaptation times for new goals compared to existing methods, showcasing the potential of our framework to advance GR in complex, real-world scenarios. By addressing the run-time limitations of traditional GR systems, this research lays the groundwork for more adaptable, efficient, automated, and accurate GR applications in a wide range of dynamic environments.

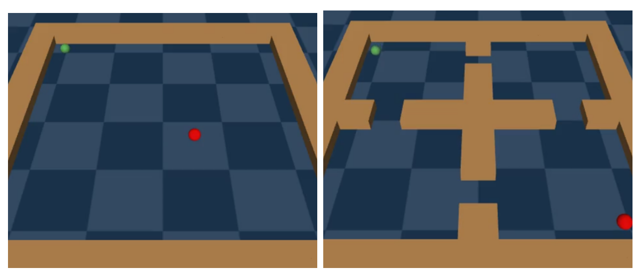

Figure 1: The Point Maze Environment features continuous state, goal, and action spaces. The green ball represents the agent in its initial state, while the red ball marks the labeled goal point. On the left, there is an empty 11x11 environment with the agent starting at 1x1 (top-left) and the goal located at 6x6 (red ball). On the right, there is a 4-rooms environment, where the agent begins at 1x1 (top-left), and the goal is at 9x9 (bottom-right, red ball).

Related Work

GR research has evolved through various methodologies. Over the past two decades, GR has been addressed primarily through symbolic approaches, such as planning (Ramírez and Geffner 2009; Sohrabi, Riabov, and Udrea

2016; Meneguzzi and Pereira 2021). These studies introduced the concept of GR by employing planners to infer the most likely goal based on given observations. However, planning-based methods require detailed domain knowledge and often rely heavily on expert inputs, which can be impractical in dynamic or unfamiliar environments. Furthermore, the emergence of stochastic and continuous domains necessitated a shift from these traditional methods toward model-free approaches.

Recent advancements have explored the integration of RL into GR tasks. Notably, the GR as RL framework (Amado, Mirsky, and Meneguzzi 2022) has shown promise by leveraging RL to infer goals without explicit domain models. However, existing methods are generally restricted to discrete state spaces. They are designed for static, single GR tasks with a finite number of possible goals, limiting their applicability in real-world settings where agents must handle continuous and evolving tasks.

To address this limitation, newer studies have reframed GR as a supervised learning problem, leveraging machine learning models to classify the most likely goal based on observations (Chiari et al. 2023; Amado et al. 2018). However, these approaches depend heavily on the availability of extensive supervised datasets and often lack interpretability, which poses challenges in critical applications where understanding the rationale behind decisions is essential.

Recent work (Shamir et al. 2024) introduced a novel problem called Online Dynamic Goal Recognition. This definition encompasses multiple GR tasks within the same domain and provides a proof of concept for implementing Dynamic GR in a discrete, simple navigational domain. The approach begins by learning Q-tables (discrete RL value functions) for a few base goals selected heuristically. For each new set of goals within the same domain, the method combines these Q-tables heuristically to quickly craft a new policy for each goal, then uses these policies to infer the most likely goal. While this work made a conceptual stride toward dynamic goal recognition, it was limited to empty navigational domains and GR tasks within the same domain.

Fang et al. (2023) expanded the scope of GR as RL using function approximations, which enabled some generalizability to new goals. However, it primarily focused on handling multiple GR tasks within the same domain without emphasizing generalizability across different tasks.

This research defines the General Dynamic Goal Recognition (GDGR) problem, proposes an algorithm for a GDGR system, and presents a specific application of this algorithm. Preliminary results demonstrate its time efficiency in adapting to new GR tasks, highlighting its potential for advancing GR in dynamic environments.

Theoretical Background

Markov Decision Process (MDP) The foundation of this research lies in the concept of a Markov Decision Process (MDP), a mathematical framework used to model decision-making in situations where both randomness and the actions of a decision-maker influence outcomes. An MDP is defined as a tuple $\mathcal{M} = (S, A, P, R, \gamma)$, where: S is the set of states; A is the set of actions; $\tau : S \times A \times S \to [0, 1]$ is the state transition probability function; $R : S \times A \to \mathbb{R}$ is the reward function. $\gamma \in [0, 1]$ is the discount factor. The objective when solving an MDP is to identify a policy $\pi : S \to A$ that maximizes the expected sum of discounted rewards.

Reinforcement Learning (RL) Building on the concept of MDPs, RL provides a practical framework for solving these problems (Kaelbling, Littman, and Moore 1996; Sutton and Barto 2018). RL is a branch of machine learning where agents learn to make optimal decisions by interacting with their environment, aiming to maximize cumulative rewards over time. RL involves learning a policy π to optimize the cumulative reward in an MDP. The value function $V^\pi(s)$ and the action-value function $Q^\pi(s, a)$ are defined as:

$$V^\pi(s) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, \pi(s_t)) \mid s_0 = s\right] \quad (1)$$

$$Q^\pi(s, a) = \mathbb{E}\left[R(s, a) + \gamma V^\pi(s') \mid s' \sim \tau(\cdot|s, a)\right] \quad (2)$$

Goal-conditioned Reinforcement Learning (GCRL) Building on the RL framework, Goal-conditioned Reinforcement Learning (GCRL) Liu, Zhu, and Zhang (2022) offers a nuanced perspective by incorporating specific goals into the learning process. This approach is central to this research as it aligns with the dynamic nature of the GR problem, where agents must adapt their policies to accommodate evolving objectives. GCRL employs a Goal-Augmented MDP (GA-MDP). A GA-MDP extends the standard MDP with an additional tuple $\langle G, p_g, \phi \rangle$, where: G represents the goal space, p_g denotes the desired goal distribution of the GA-MDP, and $\phi : S \to G$ is the mapping function that associates a state with its corresponding achieved goal. The reward function is goal-dependent and is expressed as $R : S \times G \times A \to \mathbb{R}$. The objective is to reach a goal state by using a goal-conditioned policy $\pi : S \times G \times A \to [0, 1]$ that maximizes the expected return for a goal distribution:

$$J(\pi) = \mathbb{E}_{a_t \sim \pi(\cdot|s_t, g), g \sim p_g, s_{t+1} \sim \tau(\cdot|s_t, a_t)}\left[\sum_t \gamma^t R(s_t, a_t, g)\right] \quad (3)$$

Transfer Learning in RL Transfer Learning in RL enhances learning efficiency across different domains (Zhu et al. 2023). This concept is particularly relevant to this research as it demonstrates how prior knowledge can be leveraged to address new challenges, which is crucial in GDGR.

In the context of RL, Transfer Learning involves learning an optimal policy π^* for a target domain M_t by utilizing knowledge transferred from a set of source domains M_s. This process incorporates both the experiences acquired in the source domains I_s and those from the target domain I_t, which are specific to M_t:

$$\pi^* = \arg\max_\pi \mathbb{E}_{s \sim \mu_0^t, a \sim \pi}[Q^\pi_{M_t}(s, a)], \quad (4)$$

Here, μ_0^t represents the initial distribution of states within the target environment M_t, and $Q^\pi_{M_t}(s, a)$ denotes the expected utility of selecting action a in state s, under strategy

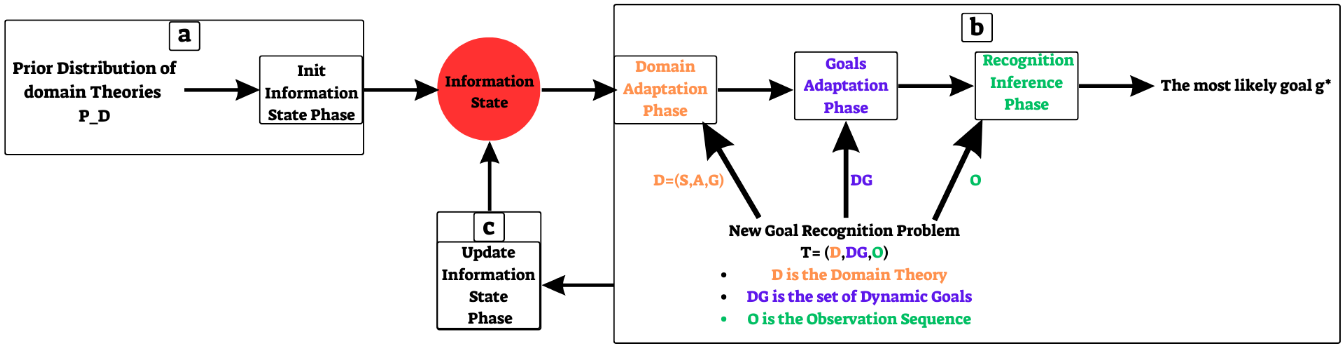

Figure 2: General Dynamic Goal Recognition Framework.

π, within M_t. The policy $\pi = \phi(I_s \sim M_s, I_t \sim M_t) : S_t \to A_t$ is tailored for M_t and is constructed using both I_t and I_s.

Goal Recognition (GR) GR is the task of inferring the most likely goal according to a series of observations. Formally defined, the GR problem can be represented as a tuple $T = (D, G, O)$, where: D represents the domain theory; G is the set of potential goals; O is a sequence of observations. The objective of GR is to identify a goal $g \in G$ that provides the best explanation for the observation sequence O. Different approaches to GR primarily differ in how they formulate the domain theory D and the techniques they use to interpret the observation sequence O.

Goal Recognition as Reinforcement Learning Amado, Mirsky, and Meneguzzi (2022) define a GR as RL problem as: $T = (D, G, O)$ where: G - the set of potential goals; O - a sequence of observations; D - the domain theory, which is divided into two types:

- Utility-based domain theory $D_Q(G)$, represented as (S, A, Q) where Q is a set of Q-functions $\{Q_g | g \in G\}$.
- Policy-based domain theory $D_\pi(G)$, represented as (S, A, π) where π is a set of policies $\{\pi_g | g \in G\}$.

General Dynamic Goal Recognition

We introduce a broader definition of the GR problem as a continuous transfer learning task, referred to as **General Dynamic Goal Recognition (GDGR)**. Given a tuple representing the prior distribution of domain theories p_0^D and a sequence of GR problems:

$$\langle p_D, (T_1 = \langle D_1, DG_1, O_1 \rangle, T_2, \ldots, T_n) \rangle,$$

each input is provided at an increasing time step, starting with p_D at time step $t = 0$. Each GR task corresponds to a distinct time step $t \in \{1, \ldots, n\}$. We define the problem of General Dynamic Goal Recognition (GDGR) as follows:

For a given series of time steps $t \in \{1, \ldots, n\}$, each time step consists of three stages of inputs, which can be given incrementally. For each time step t,

- The first input is the domain theory D_t, formally defined as $D_t = (S_t, A_t, G_t)$, where:
  - S_t is the state space of the given domain theory,
  - A_t is the action space of the given domain theory,
  - G_t is the goal space of the given domain theory.
- The second input is the set of dynamic goals DG_t used for the GR task.
- The last input is an observation sequence

  $$O_t = (\langle s_{t_1}, a_{t_1} \rangle, \langle s_{t_2}, a_{t_2} \rangle, \ldots),$$

  which might contain gaps between consecutive states and actions. Note: in an online GR setting, each state and action in the observation sequence are also provided incrementally, in the order they appear within the sequence.

The objective is to return a sequence of goals $(g_1^*, \ldots, g_n^*)$ for each time step $t \in \{1, \ldots, n\}$, where g_t^* is the recognized goal within the set of dynamic goals DG_t based on the given observations O_t and the information state IS_{t-1} carried over from the previous time step:

$$g_t^* = \arg\max_{g \in DG_t} P(g | O_t, IS_{t-1}) \tag{5}$$

Note that at time step $t = 1$, the given information state IS_0 is the initialized information state, acquired based on the prior distribution of domain theories p_D created at time step $t = 0$ (before receiving the first GR task).

Figure 2 describes the generic algorithm for the GDGR framework, outlining its different components and phases. There are three main components in the GDGR framework:

1. **Initial Information State Phase:** (line 2 – `InitInformationStatePhase` function – in Algorithm 1 and component (a) in Figure 2) Given the prior distribution of domain theories, initialize an information state that will save computation time during the subsequent phases.
2. **Specific Goal Recognition Task:** (lines 3-10 in Algorithm 1 and component (b) in Figure 2) Given the GR task and the information state:
   - Perform domain adaptation for the specific domain theory (`DomainAdaptationPhase` function in Algorithm 1) to acquire the domain knowledge required for goal adaptation and accurate recognition.

Algorithm 1: General Dynamic Goal Recognition

Require: p_D - prior distribution of domain theories

1: Init IS

2: $IS \leftarrow$ InitInformationStatePhase(p_D, IS) ▷ Initialized IS after adaptation to p_D

3: **for** all T_i in GetGoalRecognitionTask() **do**

4: Get Domain theory $D_i = \langle S_i, A_i, G_i \rangle$ from T_i

5: $IS_{D_i} \leftarrow$ DomainAdaptationPhase(D_i, IS) ▷ Domain Information State IS_{D_i} after domain adaptation

6: Get the set of new dynamic goals DG_i from T_i

7: $\{IS_g\}_{g \in GD_i} \leftarrow$ GoalsAdaptationPhase(DG_i, IS_{D_i}) ▷ Goals Information State $\{IS_g\}_{g \in GD_i}$ after goal adaptation

8: Get the Observation sequence $O_i = (\langle s_{i_0}, a_{i_0} \rangle, \dots)$ from T_i ▷ In the online GR setting, each state and action tuple is provided in a different time-step and has its own recognition inference phase

9: $g^* \leftarrow$ RecognitionInferencePhase$(\{IS_g\}_{g \in GD_i}, O_i)$ ▷ $g^* \leftarrow \arg\max_{g \in DG_i}$[DISTANCE$(O_i, IS_g)$], where DISTANCE calculates the similarity between the observation sequence and the goal Information State

10: Save and return g^*

11: $IS \leftarrow$ UpdateInformationStatePhase$(IS, IS_{D_i}, \{IS_g\}_{g \in GD_i}, T_i, g^*)$ ▷ Updated IS using the previous IS, and the current GR task, adaptations and inference

- Adapt goals for the specific set of new dynamic goals (`GoalsAdaptationPhase` function in Algorithm 1) to acquire knowledge about the new dynamic goals, enabling accurate and fast recognition.
- Conduct recognition inference (`RecognitionInferencePhase` function in Algorithm 1) for the GR task to find the most likely dynamic goal based on the given observations.

3. **Update Information State:** (line 11 – `UpdateInformationStatePhase` function – in Algorithm 1 and component (c) in Figure 2) Given the GR task with its outputs (from component (b)) and the previous information state, update the information state to include the current experience. This updated state transfers knowledge to subsequent GR tasks, enhancing their accuracy and run-time efficiency.

Algorithm 1 is a generic algorithm that can be implemented in various ways. As observed in Figure 1, where the left depicts an empty domain without obstacles and the right shows a four-rooms domain, there can be numerous scenarios of GR in the Point Maze Environment and beyond. These scenarios can be classified into three levels of abstraction:

1. There can be GR tasks assuming a small set of goals within a single domain. For a single GR task, as in the GR as RL (Amado, Mirsky, and Meneguzzi 2022) setting, there will only be a Goal Adaptation Phase, where a Q-Learning policy is learned for each goal, and the most likely goal is identified based on the similarity between the Q-Learning policy and the observation sequence.
2. There can be GR tasks with many possible goals, assuming a specific domain such as the empty Point Maze Domain. For a sequence of goals within the same domain theory, as in ODGR (Shamir et al. 2024) (a heuristic implementation for discrete simple cases), the Information State Initialization Phase involves learning a specific domain representation (a set of Q-Learning policies for heuristically chosen base goals). Then, during the Goal Adaptation Phase, heuristically crafted policies are derived using the domain representation for each new dynamic goal, followed by the same inference mechanism as in GR as RL. In this paper, we focus on this case to propose a GDGR approach for continuous goal spaces with fast adaptation times for different GR tasks within the same domain.
3. There can be GR tasks with many possible goals across multiple domains. For example, one GR task might involve the empty domain with certain dynamic goals, while another could involve the four-rooms domain with different dynamic goals. Levels 1 and 2 can be extended to this case by introducing a caching mechanism to leverage past experiences and employing a Meta-Learning Policy as the information state to enable fast adaptation to new goals and domains.

Specifically, as shown in Algorithm 2 (in the appendix), this paper focuses on cases where for all $t \in \{0, \dots, n\}$:

- All GR tasks are within the same domain theory D.
- $IS_t = \langle GCP_D, \text{cache} \rangle$, where GCP_D is the Goal-Conditioned RL Policy trained at time step $t = 0$, and the `cache` contains the buffered experiences accumulated up to this GR task at time step t (inclusive).

The challenge in such cases lies in adapting the Goal-Conditioned Policy to new dynamic goals DG_t with minimal additional learning. This iterative and continuous process improves policy adaptation mechanisms with each recognition task, exemplifying a Lifelong Learning system that efficiently adapts to new goals.

Method We implemented the generic GDGR algorithm assuming GR tasks within a specific domain (the implementation can be found in Algorithm 2 in the appendix). During the `InitInformationStatePhase`, we train a Goal-Conditioned RL (GC-RL) policy for the specific domain. This policy is utilized for all tasks. In the `DomainAdaptationPhase`, the GC-RL policy trained during the `InitInformationStatePhase` and a cache of previous GR tasks are returned.

We used sparse rewards to increase the challenge compared to dense rewards, avoiding heuristics for reward shaping that do not scale to real complex domains. To enhance

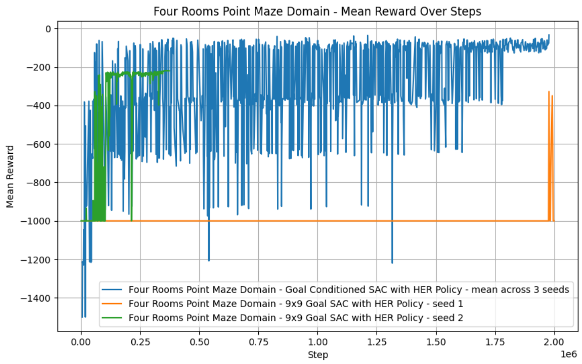

Figure 3: The figure illustrates the learning curves for the Four Rooms Point Maze Domain. The **blue curve** represents the SAC+HER Goal-Conditioned RL policy, which converged to near-optimal performance after approximately 2 million update steps. The **green curve** shows the SAC+HER goal-directed policy for the specific goal at $(9, 9)$, which converged after roughly 200,000 update steps. In contrast, the **orange curve** represents the SAC+HER goal-directed policy for the same specific goal at $(9, 9)$, which only began to converge after 2 million update steps.

learning efficiency, we combined the SAC RL algorithm with Hindsight Experience Replay (HER), which relabels unsuccessful episodes as successful for other goals encountered during training. GC-RL policies were trained by sampling random goals in the domain, while goal-directed RL policies were trained with a constant goal.

Results

In our experiments, we focused on the Point Maze Four Rooms Domain (Figure 1, right image) as described in the Appendix. We evaluate the performance and runtime of training a Goal-Conditioned RL (GC-RL) policy compared to training a goal-directed policy for individual goals. In the GC-RL scenario, the policy was trained once during the `InitInformationStatePhase` to achieve near-optimal performance for every goal in the continuous goal space. This eliminates the need for additional training or fine-tuning during subsequent GR tasks. In contrast, in the GR as RL scenario, policies are trained from scratch for every new goal. See hyperparameters details in the Appendix.

Performance Comparison

Figure 3 illustrates the training curves for the different approaches. The x-axis represents the number of update steps, and the y-axis shows the mean cumulative reward evaluated using a deterministic policy:

- The **blue curve** represents GC-RL (SAC+HER) trained with random goals per episode. After approximately 2 million update steps, the policy converges to near-optimal performance across the continuous state and goal spaces, without the need for fine-tuning.
- The **green and orange curves** represent goal-directed SAC+HER policies trained for the specific goal $(9, 9)$. The green curve (seed 1) converges to near-optimal performance after approximately 300,000 update steps, while the orange curve (seed 2) does not converge even after 2 million steps.

These results reveal clear distinctions between the GC-RL and GR as RL scenarios:

1. **Efficiency of GC-RL:** In the GC-RL scenario, a single training phase during the `InitInformationStatePhase` provides a policy capable of handling any goal in the continuous goal space without further training. This drastically reduces the overhead for new GR tasks, as only inference is required for dynamic goals.
2. **Challenges in GR as RL:** In the GR as RL scenario, each new goal requires training a separate policy from scratch. As demonstrated in the orange curve, this process can be significantly impacted by realistic sparse rewards, making it impractical in real-time applications.
3. **Inference Speed:** While the inference phase is similar for both GC-RL and GR as RL, the absence of training in GC-RL for new tasks allows for faster overall execution in dynamic environments.
4. **Sparse Rewards:** The sparse reward setting presented substantial challenges for goal-directed policies. The orange curve, which used a different random seed, illustrates a scenario where the agent struggled to converge due to limited reward signals, even with HER. In contrast, GC-RL benefited from its generalization across goals, overcoming some sparse reward issues effectively.

The comparison highlights that GC-RL significantly improves runtime efficiency and adaptability for GR tasks. While GR as RL struggles with scalability in scenarios involving continuous goal spaces, GC-RL demonstrates robustness and practicality for real-time applications.

Discussion and Conclusions

In this paper, we introduced **General Dynamic Goal Recognition (GDGR)** as a new recognition problem. This definition extends the definition of GR when goals and settings may change with time. We present an initial solution approach, implemented as a continuous transfer learning task. We demonstrated a specific application using Goal-Conditioned RL in continuous state and goal spaces, showcasing its applicability for real-time scenarios. Preliminary results using GC-RL for GR tasks validate the potential of GDGR to pave the way for real-time GR frameworks in applications such as autonomous vehicles and robotic assistants. These systems require rapid inference in dynamic environments. Our next steps are to apply GDGR in multi-domain settings by incorporating transfer learning or meta-learning techniques, e.g. to transfer a GC-RL policy from one domain to another.

References

Amado, L.; Mirsky, R.; and Meneguzzi, F. 2022. Goal recognition as reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, 9644–9651.

Amado, L.; Pereira, R. F.; Aires, J.; Magnaguagno, M.; Granada, R.; and Meneguzzi, F. 2018. Goal recognition in latent space. In *International Joint Conference on Neural Networks (IJCNN)*, 1–8. IEEE.

Bansal, G.; Nushi, B.; Kamar, E.; Lasecki, W. S.; Weld, D. S.; and Horvitz, E. 2019. Beyond accuracy: The role of mental models in human-AI team performance. In *Proceedings of the AAAI conference on human computation and crowdsourcing*, volume 7, 2–11.

Brewitt, C.; Tamborski, M.; Wang, C.; and Albrecht, S. V. 2023. Verifiable goal recognition for autonomous driving with occlusions. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 11210–11217. IEEE.

Chiari, M.; Gerevini, A. E.; Percassi, F.; Putelli, L.; Serina, I.; and Olivato, M. 2023. Goal recognition as a deep learning task: the GRNet approach. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 33, 560–568.

Fang, Z.; Chen, D.; Zeng, Y.; Wang, T.; and Xu, K. 2023. Real-Time Online Goal Recognition in Continuous Domains via Deep Reinforcement Learning. *Entropy*, 25(10): 1415.

Kaelbling, L. P.; Littman, M. L.; and Moore, A. W. 1996. Reinforcement learning: A survey. *Journal of artificial intelligence research*, 4: 237–285.

Kaminka, G.; Wendler, J.; and Ronen, G. 2001. New Challenges in Multi-Agent Intention Recognition. In *Proceedings of the Fall Symposium on Intention Recognition for Collaborative Tasks*.

Liu, M.; Zhu, M.; and Zhang, W. 2022. Goal-Conditioned Reinforcement Learning: Problems and Solutions. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, (IJCAI-22)*, 5502–5511.

Massardi, J.; Gravel, M.; and Beaudry, É. 2020. Parc: A plan and activity recognition component for assistive robots. In *IEEE International Conference on Robotics and Automation (ICRA)*, 3025–3031. IEEE.

Meneguzzi, F. R.; and Pereira, R. F. 2021. A survey on goal recognition as planning. In *Proceedings of the 30th International Joint Conference on Artificial Intelligence (IJCAI), Canadá.*

Rabkina, I.; and Forbus, K. D. 2019. Analogical reasoning for intent recognition and action prediction in multi-agent systems. In *Proceedings of the Seventh Annual Conference on Advances in Cognitive Systems*, 504–517. Cognitive Systems Foundation Cambridge.

Ramírez, M.; and Geffner, H. 2009. Plan recognition as planning. In *Twenty-First international joint conference on artificial intelligence*.

Scassellati, B. 2002. Theory of mind for a humanoid robot. *Autonomous Robots*, 12: 13–24.

Shamir, M.; Elhadad, O.; Taylor, M. E.; and Mirsky, R. 2024. ODGR: Online Dynamic Goal Recognition. *arXiv preprint arXiv:2407.16220*.

Sohrabi, S.; Riabov, A. V.; and Udrea, O. 2016. Plan Recognition as Planning Revisited. In *IJCAI*, 3258–3264. New York, NY.

Sukthankar, G.; and Sycara, K. 2011. Activity recognition for dynamic multi-agent teams. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 3(1): 1–24.

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

Towers, M.; Kwiatkowski, A.; Terry, J.; Balis, J. U.; De Cola, G.; Deleu, T.; Goulao, M.; Kallinteris, A.; Krimmel, M.; KG, A.; et al. 2024. Gymnasium: A standard interface for reinforcement learning environments. *arXiv preprint arXiv:2407.17032*.

Trick, S.; Koert, D.; Peters, J.; and Rothkopf, C. A. 2019. Multimodal uncertainty reduction for intention recognition in human-robot interaction. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 7009–7016. IEEE.

Zhang, X.; Yao, L.; Huang, C.; Sheng, Q. Z.; and Wang, X. 2017. Intent recognition in smart living through deep recurrent neural networks. In *Neural Information Processing: 24th International Conference, ICONIP, Guangzhou, China, November 14-18, 2017, Proceedings, Part II 24*, 748–758. Springer.

Zhu, Z.; Lin, K.; Jain, A. K.; and Zhou, J. 2023. Transfer learning in deep reinforcement learning: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Point Maze Domain Settings

In our experiments, we focused on the Point Maze Four Rooms Domain (Figure 1, right image) with 1000 maximum steps in an episode, with sparse rewards:

- Reward $= 0$ if the agent's Euclidean distance to the goal is less than 0.5 meters.
- Reward $= -1$ if the distance is greater than 0.5 meters.

The state space is as follows:

- x-coordinate of the green ball (`position (m)`).
- y-coordinate of the green ball (`position (m)`).
- Linear velocity in the x-direction (`velocity (m/s)`).
- Linear velocity in the y-direction (`velocity (m/s)`).

The action space includes:

- Linear force in the x-direction (`force (N)`).
- Linear force in the y-direction (`force (N)`).

The goal space comprises:

- Current goal position in the x-coordinate (`position (m)`).
- Current goal position in the y-coordinate (`position (m)`).

Hyperparameters

The following hyperparameters were used in the experiments:

- **SAC Hyperparameters:**
  - Buffer size: 1,000,000
  - Learning starts: 1,500 steps
  - Learning rate: 0.0003
  - Batch size: 256
  - Soft update coefficient (τ): 0.005
  - Discount factor (γ): 0.99
  - Training frequency: 1
  - Gradient steps: 1
- **HER Hyperparameters:**
  - Number of HER-sampled goals: 4
  - Sampling strategy: `future`

A reasonable number of hyperparameter configurations were tested, and the results were found to be similar or less efficient than those reported here.

Algorithm 2: GDGR - Implemented with Goal Conditioned Reinforcement Learning within a specific domain theory

Require: p_D - prior distribution of domain theories, assumed to be distribution with a single domain theory $D = \langle S, A, G \rangle$

1: Init $IS = \langle GCP_D, cache \rangle$

2: $IS \leftarrow$ InitInformationStatePhase(p_D, IS) ▷ Train a Goal Conditioned RL policy GCP_D for the domain theory D

3: **for** all T_i in GetGoalRecognitionTask() **do**

4: Get Domain theory $D_i = D$

5: $IS_D \leftarrow$ DomainAdaptationPhase(D_i, IS) ▷ Return the trained GCP_D from InitInformationStatePhase and the *cache*

6: Get the set of new dynamic goals DG_i from T_i

7: $\{\pi_g\}_{g \in GD_i} \leftarrow$ GoalsAdaptationPhase(DG_i, IS_D) ▷ On each goal $g \in DG_i$ it evaluates the goal specific RL policy $GCP_D(g)$ to decide whether to fine-tune it with few-shot transfer learning, or without modification (zero-shot transfer learning) $GCP_D(g)$, or taking a goal specific RL policy from the cache

8: Get the Observation sequence $O_i = (\langle s_{i_0}, a_{i_0} \rangle, \ldots)$ from T_i ▷ In the online GR setting, each state and action tuple is provided in a different time-step and has its own recognition inference phase

9: $g^* \leftarrow$ RecognitionInferencePhase($\{\pi_g\}_{g \in GD_i}, O_i$) ▷ $g^* \leftarrow \arg\max_{g \in DG_i}$[DISTANCE($O_i, \pi_g$)], where DISTANCE Calculates the similarity between the observation sequence and the goal specific RL policy

10: Save and return g^*

11: $IS \leftarrow$ UpdateInformationStatePhase($IS, GCP_D, \{\pi_g\}_{g \in GD_i}, T_i, g^*$) ▷ Caching the π_g and