Offline Reinforcement Learning-Based Human Guide Robot for Visually Impaired Navigation

Anushka Deshpande, Gil Gur-Arieh and Hadar Hai

Introduction

Example 1 *To illustrate a scenario in which an agent is assigned the task of guiding a visually impaired individual, we examine a situation where the agent (referred to as the "guide" or "robot") aims to guide the individual connected to it by a leash, and navigate through a two-dimensional obstacle-filled map towards a goal, as depicted in Figure 1. In this scenario, the figure displays the agent represented in red and the visually impaired individual represented in blue. The agent possesses complete knowledge of the map layout, including obstacles and the location of the individual, throughout all time-steps. The agent must safely maneuver the individual through all obstacles to reach the goal while ensuring they do not collide with any obstacles. Conversely, the visually impaired individual lacks visibility of the map details and can solely perceive a directional signal aligned with the leash's direction, indicating the leash's maximum extension. Due to the human-like behavior and the lack of visual perception, the individual can sometimes act irrationally, influencing the situation unpredictably. Additionally, if the leash reaches its maximum length and the individual moves in the opposite direction, he can pull the agent, but the agent cannot tug the individual. This dynamic presents unique challenges for the agent in guiding the visually impaired individual effectively.*

Related Work

A framework with Offline Reinforcement Learning utilizing human-human interaction data was shown in (Hong, Levine, and Dragan 2023). In their work, Hong et. al. showed that by learning a dataset of suboptimal human-human interactions, an agent can learn to influence a human towards better performance, by modelling and learning the underlying human latent strategy. In this work, we focus on an agent assisting a blind human to achieve a common objective, where the focus is on the limited ability of the human to sense the world.

Background

Problem Statement

We explore a scenario where an agent (guide robot) is tasked with leading a visually impaired individual through a 2D obstacle map to a designated goal region, employing a leash, akin to the example depicted in Figure 1. We formulate our problem as a single-agent *Markov Decision Process* (MDP) which is defined by $< \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma >$, where $\mathcal{S}$ are the possible states, $\mathcal{A}$ represents the actions available to the agent, $\mathcal{T}(s'|s,a)$ denotes the transition function between states given an agent's action, $\mathcal{R}(s,a,s')$ signifies the reward accumulated by the agent and γ is the reward discount factor (we use $\gamma = 0.99$).

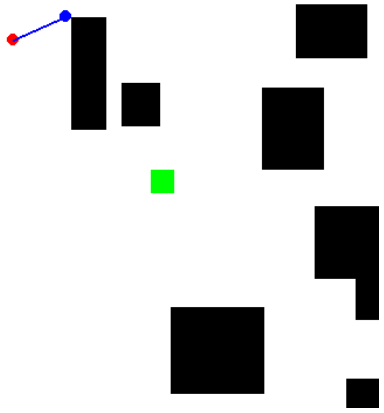

Figure 1: A simple 2D scenario featuring an agent (red) and a human (blue), connected by a leash, navigating through an obstacle-filled map towards a goal (green).

This *Reinforcement Learning* (RL) problem aims to find an agent policy $\pi(a|s)$ which maximizes the cumulative reward, in the finite-horizon formulation:

$$R(s(t_{end})) + \sum_{t=0}^{t_{end}-1} (R(s_t, \pi(s_t), s_{t+1}))$$

States

The environment is represented as a 2D grid map, incorporating the guide robot, the human, the goal, and multiple obstacles. The **state** at a given time step, s_t is defined by the positions of the goal, obstacles, robot and human. There are several mathematical ways to represent such states, as will be discussed in the Empirical Evaluation section.

Terminal states are those in which the human is at the goal or colliding with an obstacle.

Actions

The guide robot can move horizontally, vertically, or remain stationary. The action space $\mathcal{A}$ is defined as follows:

$$\mathcal{A} = \begin{cases} "U" : & (0,1), \\ "D" : & (0,-1), \\ "L" : & (1,0), \\ "R" : & (-1,0), \\ "S" : & (0,0) \end{cases}$$

Where each line represents a different action the agent can take, representing "up", "down", "left", "right", "stationary" accordingly.

State-Action Transition Function

In our problem formulation, we assume the human can move in a similar manner to the guiding robot (up, down, left, right, stationary) and independently from the robot. It is important to emphasize that we formulate the human's actions as a part of the stochastic environment, as they are not directly controlled by the robot actions. Therefore, given a state $s \in \mathcal{S}$ and the robot's chosen action $a \in \mathcal{A}$, the subsequent state $s' \in \mathcal{S}$ (new positions) will contain the new human and robot position, where the relation between the new human position and the robot's action (a) is unknown and we assume it cannot be modeled directly:

$$\mathbf{h}_{t+1} = \mathbf{h}_t + \begin{cases} (0,1) \\ (0,-1) \\ (1,0) \\ (-1,0) \\ (0,0) \end{cases}$$

Where $\mathbf{h}_t$ denotes the human position (x, y) in time-step t.

The new robot position is defined by the robot action:

$$\mathbf{r}_{t+1} = \mathbf{r}_t + \begin{cases} (0,1) & a = "U" \\ (0,-1) & a = "D" \\ (1,0) & a = "L" \\ (-1,0) & a = "R" \\ (0,0) & a = "S" \end{cases}$$

Where $\mathbf{r}_t$ denotes the robot position (x, y) in time-step t. However, the above relation is only true unless one of the following conditions holds:

- The robot tries to move outside of map boundaries. In that case, $\mathbf{r}_{t+1} = \mathbf{r}_t$.
- The robot tries to move into an obstacle. In that case, $\mathbf{r}_{t+1} = \mathbf{r}_t$.
- The leash is at its maximum length and the robot tries to move away from the human. In that case, $\mathbf{r}_{t+1} = \mathbf{r}_t$.
- The leash is at its maximum length and the human moves away from the robot (the human drags the robot away). In that case, $\mathbf{r}_{t+1}$ will be the closest location to $\mathbf{r}_t$ which satisfies the euclidean distance between the human and the robot is no greater than the maximum leash size. Mathematically,

$$\mathbf{r}_{t+1} = \arg\min_{\mathbf{r}} \|\mathbf{r} - \mathbf{r}_t\|$$

Subject to the condition:

$$\|\mathbf{r}_{t+1} - \mathbf{h}_{t+1}\| \leq \text{max\_leash\_size}$$

Reward

The reward accumulated by the agent at each step is the weighted sum of the following components:

- $R_{human\_col}(s)$ - Negative reward for human collision with obstacle
- $R_{robot\_col}(s)$ - Negative reward for robot collision with obstacle
- $R_{goal}(s)$ - Positive reward for reaching the goal region
- $R_{step}(s)$ - Negative fixed reward for each time-step elapsed

YOUR METHOD

Imitation Learning Overview

Imitation Learning aims to replicate the decision-making process of a proficient agent without explicitly modeling the dynamics of the environment. It is particularly advantageous in situations where safe and efficient behavior is critical, and exploration risks are undesirable.

Baseline Methodology

Setting a Baseline with Behavior Cloning

To establish a robust foundation for our guide robot's navigation capabilities, we begin by setting a baseline using Behavior Cloning (BC). BC serves as our initial approach to learning from expert demonstrations, which is particularly suited for applications requiring high levels of safety and operational efficiency.

Given a dataset $\mathcal{D} = \{(s_i, a_i)\}_{i=1}^{M}$ of state-action pairs collected from expert demonstrations, our goal is to learn a policy π_{BC} that closely mimics the expert's policy π^*. This is achieved by optimizing the policy to minimize the discrepancy between the actions predicted by π_{BC} and the actions a_i^* demonstrated by the expert, across all observed states s_i in $\mathcal{D}$.

The optimization objective for BC can be formalized as follows:

$$\pi_{\text{BC}}^* = \underset{\pi \in \Pi}{\text{argmin}} \sum_{i=1}^{M} \ell(\pi(s_i), a_i^*), \tag{1}$$

where ℓ denotes the loss function tailored to either discrete or continuous action spaces. For discrete actions, the Negative log-likelihood (NLL) loss is commonly used:

$$\ell(\pi(s), a^*) = -\ln \pi(a^*|s), \tag{2}$$

while for continuous actions, the square loss is preferred:

$$\ell(\pi(s), a^*) = \|\pi(s) - a^*\|_2^2. \tag{3}$$

By focusing on minimizing this loss, BC provides a straightforward and effective method for transferring expert knowledge to the guide robot, establishing a strong baseline for its navigation strategy. This approach ensures that the robot learns to navigate through environments in a manner that is both safe for the visually impaired individual and closely aligned with the expert's proven strategies.

Advantage-Filtered Behavioral Cloning (AFBC)

While Behavioral Cloning (BC) provides a foundational approach for learning from expert demonstrations, its effectiveness is limited when the dataset includes suboptimal experiences. This limitation arises from BC's inherent strategy to indiscriminately mimic all observed actions, including those that may lead to inferior outcomes. To address this challenge, we introduce Advantage-Filtered Behavioral Cloning (AFBC)(Grigsby and Qi 2021), an extension of BC that selectively replicates only those actions deemed advantageous based on a comparison metric known as the advantage function.

The Concept of Filtering in BC

The pivotal innovation of AFBC hinges on leveraging the advantage function, defined as $A^\pi(s,a) = V^\pi(s,a) - V^\pi(s)$, to measure the expected gain of executing action a in state s compared to the default value of the state s under the existing policy. By applying this measure, AFBC refines Behavior Cloning (BC) into a process that selectively imitates actions promising superior outcomes, thereby enhancing its decision-making criterion.

Estimating the Advantage Function

The AFBC estimates the advantage function as follows:

$$\hat{A}^\pi(s,a) = V^\phi(s,a) - \frac{1}{k}\sum_{i=0}^{k} V^\phi(s), \tag{4}$$

where $a'^{(i)} \sim \pi_\theta(s)$ are actions sampled from the current policy, and Q^ϕ denotes the learned Q-function, approximated using techniques from off-policy actor-critic methods. This estimation process enables a practical approach to identifying advantageous actions without the need for true value functions.

Filtering Mechanism

With a reliable estimate of the advantage function in hand, AFBC incorporates a filtering mechanism into the traditional BC loss, allowing the model to focus on beneficial actions:

$$L_{\text{actor}} = \mathbb{E}_{(s,a)\sim\mathcal{D}}\left[-f(\hat{A}^\pi(s,a))\log\pi_\theta(a|s)\right], \tag{5}$$

where $f(\hat{A}^\pi(s,a))$ represents the filtering function applied to the estimated advantage. A commonly used filter is the exponential filter:

$$f(\hat{A}(s,a)) = \exp(\beta\hat{A}^\pi(s,a)), \tag{6}$$

with β serving as a hyperparameter to control the influence of advantage estimates. This method not only preserves actions with positive advantages but also adjusts their contribution based on the magnitude of their estimated benefits, thereby enhancing the model's ability to prioritize and learn from the most valuable demonstrations.

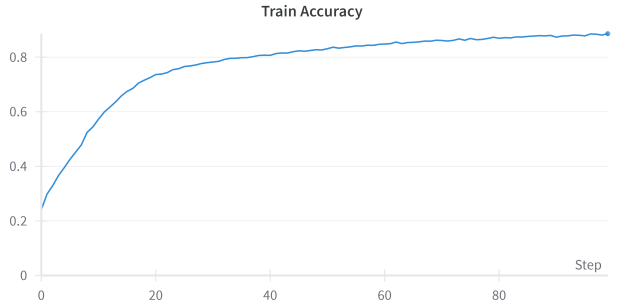

Figure 2: Training accuracy of the AFBC agent .

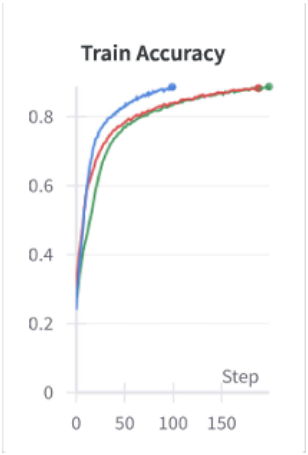

Figure 3: Training accuracy while increasing size of value network input-128 (blue), 256 (red), and 512 (green)

Training Results of the AFBC Agent

The AFBC agent demonstrated a strong learning capacity during the training phase. Figure 2 shows the training accuracy, which reached 84%, reflecting the agent's proficiency in learning from the training dataset.

Model Improvements

Upon increasing the dimensions of the value network, we were able to capture a more intricate representation of the state space. During our experimentation, the input dimensionality of the value network was systematically varied to examine its impact on the ABC agent's learning performance. We employed input dimensions of 128 (blue), 256 (red), and 512 (green), each represented by a distinct color in Figure 3.

Our findings indicate that the model with the smallest input dimension of 128 experienced early stopping in the training process, unable to continue improving due to the limited capacity to capture the complexity of the data. As we increased the dimensions to 256 and 512, the learning continued for a more extended period, and the models exhibited better performance and higher training accuracy.

These results indicate that the ABC agent not only learns effectively but can also be further optimized by architectural

adjustments, such as increasing the dimension of state representation

Empirical Evaluation

Dataset

Human-Human data collection The data utilized for all offline Reinforcement Learning algorithms was created using human-human interactions in the created environment. Two humans were able to play the game simultaneously, so that the "guide" player was able to see the entire screen (obstacles, goals, players, leash) while the "human" player was only able to see a small screen which showed either a line pointing in the direction of the leash (if the leash was taught), or nothing.

A total of 100 games (episodes) were played. All games terminated when the "human" player either reached the goal or collided with an obstacle ("guide" player collision with obstacle yields negative reward but is not terminal).

Pre-processing and representation of data After the human-human data was collected, the data was pre-processed in order to fit to the algorithm and model.

The original data had time frames in which both players were idle and then "suddenly" chose an action (when observing the screen or thinking about how to proceed, for example). While this behavior is sensible for human players, it causes the guide behavior policy (π_β) to be non-Markovian (No change in state but change in consecutive action). To avoid this, such lines were removed from the data.

While training the various agents, we found that different representations of the state affect both the training accuracy of the behavior cloning agents. State representations that were tried are:

1. Positions array - includes positions of the obstacles, goal and players at every timestep:
   (a) In the "world" reference frame (x, y values of the objects)
   (b) In the agent ("guide" player - reference frame (relative position vectors $(\Delta x, \Delta y)$ between the objects and the player
2. Pixel images that interpret RGB values as the players, goal and obstacles (as seen in figure 1).
3. Lidar-like beam vector - at every timestep 360 rays are "cast" from the agent and return distances measured until the other player, goal or obstacle detected.

Of all the representations above, the "Lidar"-like beam representation showed best results, both in NN test accuracy as well as qualitative agent behavior. The pixel images representation increased data size significantly, and a resize of the images was necessary to run the training on a standard CPU.

Setup

The evaluation of the agent performance was performed directly on the environment. To save time, a "human-imitation" agent was constructed; This agent makes a random action at a probability of ϵ and otherwise makes N steps towards a taught leash or stays still if the leash is not at maximum length.

Three agents were tested:

- Baseline agent - assumes it is at the same position as the "human" player and moves towards the goal.
- Behavior cloning (BC) agent.
- Advantage-filtered behavior cloning (AFBC) agent.

The agents were tested in maps of the same size as the data-collection map, with either no obstacles or 1 obstacle (random goal and initial player positions at every game). A step-count constraint was added so that episodes that do not converge to reach a goal end in failure. Each agent played a total of 200 games, where a game is called "successful" if the "human" player reaches the goal safely.

Results

The success rates of all 3 agents are shown in table 1.

	Baseline	BC	AFBC
0-obs.	99	74	83
1-obs.	86	64	69

Table 1: Success rate (%) for the baseline goal-follow, Behavior Cloning (BC) and Advantage-Filtered Behavior Cloning (AFBC), for obstacle-free or 1-obstacle maps

Discussion

In an obstacle-free (or nearly obstacle-free) map, it is sensible that the baseline agent which moves directly in the direction of the goal has more success in reaching the goal. The results clearly show better performance of the Advantage-Filtered Behavior Cloning agent, even in the presence of an obstacle.

Both agents (BC and AFBC) occasionally showed "policy loops", in which the agent actions cause 2 or 3 states to repeat, often near the goal, which led the players close to the goal but the episode to end in "failure".

Conclusion

In this work, we developed a simple but versatile environment which can be utilized in future research projects for human-human data collection, implementation and evaluation of various guiding agents (RL or otherwise), as well as a multi-agent reinforcement learning (MARL) setup.

Representation of the data (states) has proved to be a key component in training behavior cloning agents. The "LidAR"-like beam method has shown best results while keeping the data size relatively compact.

The addition of value-function based weights to the behavior-cloning policy training has empirically shown to improve agent performance.

We acknowledge the limitations of this work, specifically the limitation of data (only 100 human-human games) and of the environment parameters (number of obstacles, goal size, leash size, etc.). For future work, we recommend examining the effect of those environment parameters on results. We

also believe other offline reinforcement learning (RL) can be utilized to obtain an even better agent performance, as the behavior-cloning algorithms rely on the "optimality" of the data-collection policy π_β.

Git Repository Link

https://github.com/hadar-hai/GuideRobotOfflineRL.git

References

Grigsby, J.; and Qi, Y. 2021. A closer look at advantage-filtered behavioral cloning in high-noise datasets. *arXiv preprint arXiv:2110.04698*.

Hong, J.; Levine, S.; and Dragan, A. 2023. Learning to Influence Human Behavior with Offline Reinforcement Learning. arXiv:2303.02265.