Guided Data Augmentation for Offline Reinforcement Learning and Imitation Learning

Nicholas E. Corrado[1], Yuxiao Qu[2], John U. Balis[1], Adam Labiosa[1], Josiah P. Hanna[1]

***Abstract*— Learning from demonstration (LfD) is a popular technique that uses expert demonstrations to learn robot control policies. However, the difficulty in acquiring expert-quality demonstrations limits the applicability of LfD methods: real-world data collection is often costly and the quality of the demonstrations depends greatly on the demonstrator's abilities and safety concerns. A number of works have leveraged data augmentation (DA) to inexpensively generate additional demonstration data, but most DA works generate augmented data in a random fashion and ultimately produce highly suboptimal data. In this work, we propose Guided Data Augmentation (GuDA), a human-guided DA framework that generates expert-quality augmented data. The key insight of GuDA is that while it may be difficult to demonstrate the sequence of actions required to produce expert data, a user can often easily identify when an augmented trajectory segment represents task progress. Thus, the user can impose a series of simple rules on the DA process to automatically generate augmented samples that approximate expert behavior. To extract a policy from GuDA, we use off-the-shelf offline reinforcement learning and behavior cloning algorithms. We evaluate GuDA on a physical robot soccer task as well as simulated D4RL navigation tasks, a simulated autonomous driving task, and a simulated soccer task. Empirically, we find that GuDA enables learning from a small set of potentially suboptimal demonstrations and substantially outperforms a DA strategy that samples augmented data randomly.**

I. Introduction

Learning from demonstration (LfD) is a popular learning paradigm in which robots learn to solve complex tasks by leveraging successful demonstrations provided by a human. In contrast to more traditional control methods that require a human expert to pre-program desired control sequences or formulate control as a constrained optimization problem [1]–[3], LfD is an intuitive alternative that enables experts and non-experts alike to develop control policies. Instances of LfD such as imitation learning (IL) [4] and offline reinforcement learning[1] (RL) [9] have proven to be viable methods for learning effective policies in real-world tasks such as robot manipulation [10]–[12] and autonomous driving [13], [14].

The performance and generalization capabilities of LfD methods depends greatly on the quantity and quality of demonstrations provided to the learning agent [15]. Ideally, we would provide large amounts of expert-quality demonstrations, but acquiring such data is often challenging in real-world tasks: the expense of data collection often limits us to just a few demonstrations, and the quality of these demonstrations depends on the demonstrator's level of expertise as well as the degree of safety they must exercise while collecting data. Moreover, while prior works have shown that many offline RL algorithms perform well even with highly suboptimal data [5]–[8], these same works show that offline RL performs far better with expert-quality data. As such, we focus on developing methods that enable practitioners to cheaply acquire high-quality demonstrations.

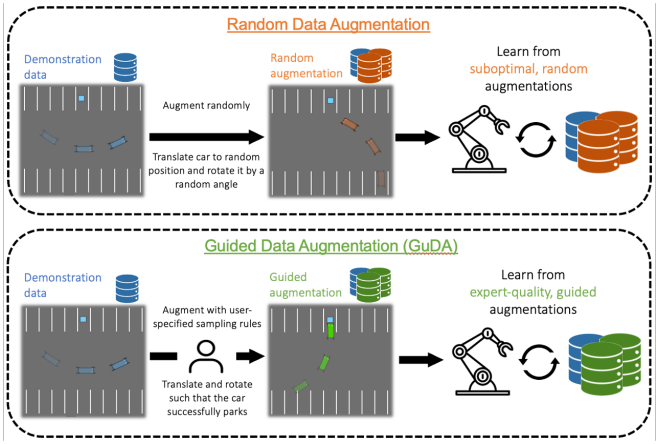


Fig. 1: An overview of GuDA applied to a parking task. The blue square indicates the goal parking spot.

In this work, we introduce **Gu**ided **D**ata **A**ugmentation (GuDA), a human-guided data augmentation framework capable of generating large amounts of expert-quality data from a limited set of demonstrations. Data augmentation (DA) refers to techniques that generate additional synthetic experience without the expense of task interaction by applying transformations on previously collected experience. These transformations – or *data augmentation functions* (DAFs) – typically leverage task-specific invariances and symmetries inherent to many real-world tasks (*e.g.* translational invariance [16], [17], gait symmetry [18], [19]). Most prior DA works sample augmented data from a given DAF uniformly at random [16], [20]–[23] or randomly generate augmented trajectories from a learned dynamics model [24]–[26]. Unfortunately for the use of these techniques for LfD, randomly generated augmented experience is generally highly suboptimal and does not capture behaviors needed to solve a given task. The key insight of GuDA is that a human expert can often determine if a trajectory segment resembles expert data by simply checking if its sequence of states brings the agent closer to solving the task. Thus, instead of

[1]With the University of Wisconsin — Madison, Department of Computer Sciences {ncorrado, balis, labiosa, jphanna}@wisc.edu

[2]With Carnegie Mellon University, Department of Computer Science yuxaioq@andrew.cmu.edu.

[1]Offline RL can learn from suboptimal data, but it is far more successful with expert demonstrations [5]–[8]. Thus, we view it as an LfD method.

randomly sampling augmented data, GuDA uses a series of user-defined rules to automatically generate augmented data that makes substantial progress towards task completion.

To make this concept more concrete, imagine we are training an autonomous vehicle to park in a parking lot using a limited set of demonstrations (Fig. 1). Since a parking lot has a relatively uniform surface, we can generate augmented experience by translating and rotating the agent in our demonstrations. Expert behaviors for this tasks include (1) driving towards the desired parking spot and (2) orienting the car inside the parking spot. However, a uniformly random sampling of augmented data will most often produce data in which the agent drives away from the parking spot or approaches it at an unfavorable angle. With GuDA, we can generate relevant augmented data by only sampling augmented trajectory segments in which the agent successfully parks the car. Such augmented data closely mimics data that an expert policy would generate and provides more varied expert-quality data without asking for more demonstrations.

The benefits of GuDA are twofold: First, GuDA enables practitioners to generate expert data without the expense of task interaction. Second, instead of requiring that an expert demonstrate an optimal sequence of actions required to solve a task, GuDA simply requires the user to judge if an augmented trajectory segment represents progress towards task completion. We evaluate GuDA with off-the-shelf offline RL and behavior cloning algorithms on simulated navigation, autonomous driving, and soccer tasks as well as a physical robot soccer task. Empirically, GuDA enables robots to learn effective policies starting from just a few demonstrations – even highly suboptimal demonstrations. Moreover, we find that GuDA greatly outperforms an DA strategy that samples augmented data uniformly at random. In summary, our contributions are

1) We demonstrate how a user can guide data augmentation to inexpensively produce expert-quality data from potentially suboptimal experience.
2) We show that GuDA significantly outperforms the most widely used DA strategy – one that samples augmented data uniformly at random – highlighting the benefits of a more intentional approach to DA. In fact, this random DA strategy often *harms* performance.

II. Related Work

In this section, we provide an overview of prior work in LfD and data augmentation.

A. Learning from Demonstrations (LfD)

LfD methods have taken many forms in the literature. In this section, we discuss LfD methods relevant to our work.

1) Imitation Learning: The simplest imitation learning (IL) method is behavior cloning (BC), a technique in which the agent learns to map observed states to expert actions in a supervised manner. BC often produces policies that generalize poorly to unobserved states and cannot produce policies that exceed the performance level achieved by the expert [27]–[29]. DAgger [27] mitigates these drawbacks by iteratively running BC and then collecting additional data with the BC-trained policy, though this online interaction may be prohibitively expensive in robotics tasks [30].

In contrast to BC, inverse RL (IRL) methods [31] infer a reward function from demonstrations and then learn a policy which optimizes this reward function. By avoiding simple copying of the demonstrator, the agent can generalize to states not provided in demonstrations and potentially exceed the demonstrator's performance. However, IRL assumes the demonstrator optimizes some true reward function and thus still requires expert data. Moreover, many IRL algorithms require online interaction with the task and thus, like DAgger, may be impractical when further online interaction is infeasible [32]–[35]. To address limitations found in both types of IL methods, GuDA generates large amounts of expert data from a limited set of demonstrations.

2) Offline Reinforcement Learning: Offline RL [9] is a learning paradigm in which an RL agent learns from a static dataset of task demonstrations. Rather than mimicking demonstrations, these methods learn a reward-maximizing policy from reward labels provided with the demonstrations. These methods are designed such that, in principle, they can learn even with suboptimal data. Nevertheless, offline RL is generally far more successful with expert data [5]–[8]. Thus, we view offline RL as an LfD technique.

One core challenge with offline RL is extrapolation error: state-action pairs outside of the dataset's support can attain arbitrarily inaccurate state-action values during training, causing learning instabilities and poor generalization during deployment [36]. This challenge is particularly problematic for real-world robotics tasks in which offline data is scarce. Offline RL algorithms typically combat extrapolation error with policy parameterizations that only consider state-action pairs within the dataset [6], [37], [38] or behavioral cloning regularization [8], [39], [40]. GuDA, like other DA strategies (Sec. II-B), can be viewed as a technique to mitigate extrapolation error by simply generating more data without further task interaction. However, GuDA also improves dataset quality by generating expert augmented data.

B. Data Augmentation

Data augmentation (DA) refers to techniques which generate synthetic data by transforming previously collected experience. DA has been applied a variety of tasks, including algorithm discovery [41], locomotion [18], [19], and physical robot manipulation [42], [43]. This technique is particularly useful for robotics; it can generate data that matches real-world dynamics without further task interaction.

DA is most often used to generate perturbed data with the same semantic meaning as the original data. Many vision-based RL works have trained agents to be robust to visual augmentations commonly used in computer vision [44]–[51], and similar approaches have been applied to to non-visual tasks [52]–[54]. These approaches are orthogonal to GuDA; they use DA to learn robust policies, whereas GuDA uses DA to improve dataset quality. Perturbation-based DA methods

more closely relate to domain randomization [55]–[57] which also aims for policy robustness.

Other works exploit invariances and symmetries in a task's dynamics to generate data that is semantically different from the original data. Hindsight experience replay (HER) [58]–[62] counter-factually relabels a trajectory's goal. Counterfactual Data Augmentation (CoDA) [16] and Model-based CoDA (MoCoDA) [17] generate additional data by stitching together locally independent features of different transitions. Several works use a learned model to generate augmented data [23]–[25], [63]–[66]. Most of these works focus on developing DAFs or frameworks for incorporating augmented data into learning and simply generate augmented experience in a random fashion. In contrast, GuDA focuses on the importance of sampling expert-quality augmentations.

Two prior works are most closely related to GuDA: EXPAND [51], which applies visual augmentations to irrelevant image regions identified by human feedback, and MoCoDA [17], which allows users to specify a *parent distribution* to control the distribution of augmented data. GuDA differs from EXPAND in that we focus on non-visual tasks with more complex DAFs more relevant to robotics. In contrast to MoCoDA, GuDA is a model-free DA framework and can be applied when data is too scarce to model the data distribution, as is commonly the case in physical tasks. Moreover, GuDA provides a more intuitive interface for DA that enables fine-grained control of the distribution of augmented data.

III. Preliminaries

In this section, we formalize the RL setting and the notion of a data augmentation function that we use in this work.

A. Offline Reinforcement Learning

Since our empirical analysis considers offline RL methods, we adopt notation from the RL literature and formalize a task as a sequential decision-making process with a known reward function. We note that a reward function may be unavailable for certain tasks; in such case, our proposed GuDA framework can use BC instead. When one is available, we can use offline RL to attempt to improve over the demonstrator.

Formally, we consider finite-horizon Markov decision processes (MDPs) [67] defined by $(\mathcal{S}, \mathcal{A}, p, r, d_0, \gamma)$ where $\mathcal{S}$ and $\mathcal{A}$ denote the state and action space, respectively, $p(\boldsymbol{s}' \mid \boldsymbol{s}, \boldsymbol{a})$ denotes the probability density of the next state $\boldsymbol{s}'$ after taking action $\boldsymbol{a}$ in state $\boldsymbol{s}$, and $r(\boldsymbol{s}, \boldsymbol{a})$ denotes the reward for taking action $\boldsymbol{a}$ in state $\boldsymbol{s}$. We write d_0 as the initial state distribution, $\gamma \in [0, 1)$ as the discount factor, and H the length of an episode. We consider stochastic policies $\pi_\theta : \mathcal{S} \times \mathcal{A} \to [0, 1]$ parameterized by θ. The RL objective is to find a policy that maximizes the expected sum of discounted rewards $J(\theta) = \mathbb{E}_{\pi_\theta, \boldsymbol{s}_0 \sim d_0}\left[\sum_{t=0}^{H} \gamma^t r(\boldsymbol{s}_t, \boldsymbol{a}_t)\right]$. In the offline RL paradigm, the agent cannot collect data through environment interaction and must instead learn from a static dataset $\mathcal{D}$ of transitions collected by a different policy.

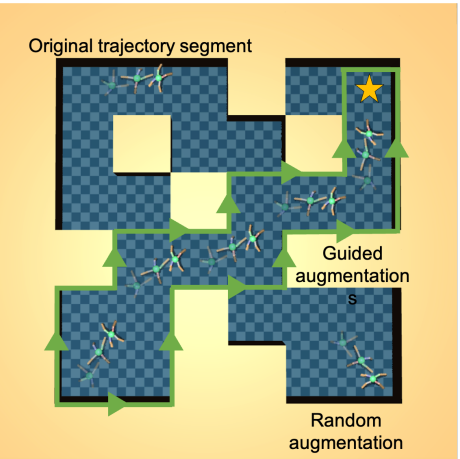


Fig. 2: An illustration of guided data augmentation (GuDA) in a locomotion task. GuDA translates and rotates a provided trajectory segment (top left) to generate a demonstration of the agent walking towards the goal (gold star). Randomly sampled augmented data (bottom right) may have highly suboptimal behavior that leads the agent away from the goal.

B. Data Augmentation Functions

In this section, we formally introduce a general notion of a data augmentation function (DAF). At a high level, a DAF generates augmented data by applying transformations to an input trajectory segment. These transformations often exploit task-specific invariances and symmetries relevant to many real-world tasks. More formally, let $\mathcal{T}$ denote the set of all possible trajectory segments and let $\Delta(\mathcal{T})$ denote the set of distributions over $\mathcal{T}$. A DAF is a stochastic function $f : \mathcal{T} \to \Delta(\mathcal{T})$ mapping a trajectory segment $((\boldsymbol{s}_i, \boldsymbol{a}_i, r_i, \boldsymbol{s}'_i))_{i=1}^k$ of length k to an augmented trajectory segment $((\tilde{\boldsymbol{s}}_i, \tilde{\boldsymbol{a}}_i, \tilde{r}_i, \tilde{\boldsymbol{s}}'_i))_{i=1}^k$. We assume DAFs always assign the true reward to augmented transitions, *i.e.*, $\tilde{r} = r(\tilde{\boldsymbol{s}}, \tilde{\boldsymbol{a}})$. As in most prior works, we assume a user can specify a DAF f that exploits a symmetry or invariance in a given domain [16], [58], [68].

IV. Guided Data Augmentation

The difficulty in acquiring near-optimal demonstrations limits the applicability of LfD methods. While DA can inexpensively generate data from a limited set of prior data, most of the resulting augmented data is not expert-quality. To make augmented data more relevant for imitation learning and offline RL, we introduce Guided Data Augmentation (GuDA), a DA framework that uses a set of user-specified rules to automatically generate augmented data that resembles expert data. This approach thus shifts the burden from the user having to provide optimal actions for demonstrations to the user simply having to understand which augmented data represents progress towards task completion.

A. Method Overview

We assume access to a dataset $\mathcal{D}$ of demonstrations and a task-relevant DAF f from which we can sample augmented data. Prior to offline training, GuDA generates an augmented

Task	Task Description	Initial Dataset Size	GuDA Sampling Rules (τ = input trajectory segment)
maze2d-umaze maze2d-medium maze2d-large	A force-actuated point-mass must navigate to a fixed goal from a random initial position. The agent receives +1 reward at the the goal and 0 otherwise.	1500 (5 trajectories) 3000 (5 trajectories) 4000 (5 trajectories)	`Translate` τ to a random maze position, and then `Rotate` τ such that the agent moves along the shortest path to the goal.
antmaze-umaze antmaze-medium antmaze-large	A quadruped must navigate to a fixed goal from a fixed initial position. The agent receives +1 reward at the goal and 0 otherwise.	818 (1 trajectory) 2754 (2 trajectories) 4685 (5 trajectories)	`Translate` τ to a random maze position for the agent moves along the shortest path to the goal.
parking	An autonomous vehicle must park front-first into a designated parking spot. The agent receives a dense reward based on its distance to the parking spot and how closely the car aligns with the spot.	302 (10 trajectories)	First, use `RelabelGoal` to change τ's goal to randomly sampled goal (parking spot). Then, `Translate` τ such that the agent's final position is at the goal, and `Rotate` τ such that the car is within the parking spot.
soccer-sim	An agent must kick a ball to a fixed goal location. Agent and ball positions are initialized randomly. The agent receives reward based on its distance to the ball and the ball's distance to the goal.	1500 (3 trajectories)	`Translate` τ such that the ball's final position is at the goal, and then `Rotate` τ randomly such that τ remains in-bounds. Afterwards, `Reflect` τ with probability 0.5.

TABLE I: Simulated tasks and GuDA sampling rules. Section V-C details sampling rules for our physical robot soccer task.

dataset $\tilde{\mathcal{D}}$ consisting of the original demonstrations plus n augmented samples generated by f. Afterwards, an agent learns from $\tilde{\mathcal{D}}$ using an off-the-shelf LfD algorithm. The core difference between GuDA and previous DA works (*e.g.*, [16], [44]) lies in how GuDA samples augmented data from f. In general, most augmentations capture highly suboptimal behaviors. Instead of sampling augmented data uniformly at random as in commonly done in prior works, GuDA imposes a series of simple *sampling rules* to automatically generate expert-quality augmented data. A user can often identify such sampling rules using basic intuitions on how to solve a task.

To illustrate how a user might identify sampling rules, consider a maze navigation task in which legged robot must reach a fixed goal state from a fixed initial position (Fig. 2). In this task, we assume access to a DAF that translates the agent to a new position and rotates the direction in which the agent moves. While it is difficult to demonstrate the precise sequence of leg movements required to optimally solve the maze, we can easily identify when an augmented version of an existing trajectory segment progresses the agent towards the goal. A randomly sampled augmentation from our translate-and-rotate DAF will most likely have the agent move *away* from the goal rather than towards it. Moreover, the agent only needs to learn suitable actions for a small fraction of maze positions near the shortest path to the goal, but our DAF will mostly generate data in regions of the maze that an optimal policy would never visit. To ensure we generate expert augmented data, we can simply restrict our DAF to (1) only sample new positions near the shortest path to the goal (green region), and (2) always rotate the agent so its displacement is closely aligned with the shortest path (green arrows).

The exact specification of sampling rules for GuDA is a domain-specific process that depends on which DAFs are available as well as what task progress looks like in a given domain. In this work, we focus on navigation, manipulation, and autonomous driving tasks which have intuitive notions of task progress; an agent makes progress if it moves closer to a specified goal location (navigation and driving) or if it moves an object closer to a specified goal location (manipulation). In the remainder of this section, we describe the DAFs we use and discuss how we can sample from these DAFs to ensure we only generate data that shows task progress.

B. Implementation

We focus on four DAFs that leverage invariances and symmetries common to many tasks in the physical world:

1) **`Translate`:** Since the dynamics of agents and objects are often independent of their position, this DAF translates the agent and/or object to a new position.
2) **`Rotate`:** Since the dynamics of agents and objects are often independent of their orientation, we can rotate the direction the agent and/or object faces to produce motion in a different direction.
3) **`Reflect`:** An agent that moves to the left often produces a mirror image of an agent moving to the right, so we can reflect the agent's left-right motion.
4) **`RelabelGoal`:** In goal-conditioned tasks, dynamics are generally independent of the desired goal state [58]. Thus, we can replace the true goal with a new goal.

Table I describes the tasks in our empirical analysis as well as the sampling rules we implement to automatically generate expert-quality data from combinations of these DAFs. We include the following simulated tasks: D4RL maze2d and antmaze locomotion tasks [69], a parking task [70], and a robot soccer task. We also validate GuDA on a physical robot soccer task, and we further discuss this task's sampling rules in Section V-C.

GuDA can in principle be implemented in many different ways and can be adapted depending on which DAFs are available. For instance, we found that the `Rotate` DAF was helpful in maze2d but often harmed performance in antmaze. Thus, in antmaze, we simply translate trajectory segments to relevant positions for which the original displacement direction represents significant task progress. Since offline RL methods perform better with noisy expert data [28], we inject noise into our sampling rules. For instance, in maze2d tasks, all rotated trajectory segments align closely – but often not exactly – with the optimal direction of motion.

V. Experiments

We design an empirical study to answer the following questions: (1) Does GuDA enable learning from a limited set

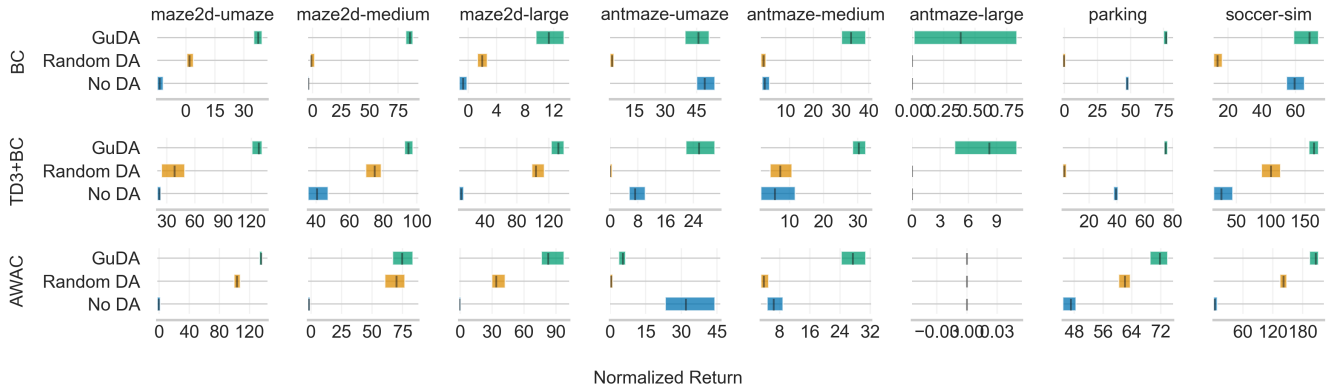


Fig. 3: We report IQM normalized returns over 10 independent runs with 95% bootstrapped confidence intervals for offline learning with GuDA, Random DA, and No DA using different algorithms. We compute normalized returns computed as $\texttt{normalized return} = 100 \cdot \frac{R - \texttt{random return}}{\texttt{expert return} - \texttt{random return}}$ where $\texttt{expert return}$ and $\texttt{random return}$ denote the return of the expert demonstrator and a policy that chooses actions uniformly at random, respectively.

of potentially suboptimal demonstrations? (2) Does GuDA yield larger returns than a random DA strategy?

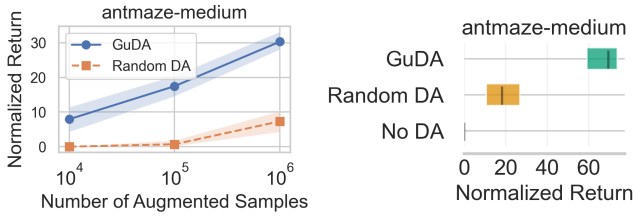
A. Empirical Setup

We first evaluate GuDA on simulated tasks described in Table I. In all tasks, we start with a small initial dataset containing at least one successful – though not necessarily expert-level – demonstration (Table I). These datasets often contain failures or suboptimal behaviors as well: maze2d datasets contain data in which the agent moves away from the goal, soccer datasets contain trajectories where the agent kicks the ball out of bounds, and parking datasets contain trajectories where the car fails to park at its designated goal. For maze2d and antmaze tasks, we hand-pick a small number of trajectory segments from the original '-v1' and '-diverse-v1' D4RL datasets, respectively. For the remaining tasks, we use pre-trained expert policies to generate demonstrations.

We consider two baselines: a DA strategy that randomly samples augmented data (Random DA), and no augmentation (No DA). We generate 1 million augmented transitions and then perform offline learning with BC, TD3+BC [8], and AWAC [39]. We train for 1 million policy updates and report the inter-quartile mean (IQM) return achieved over 10 independent runs [71].

B. Simulated Experiments

Fig. 3 shows IQM normalized returns for each algorithm in each task. GuDA almost always outperforms Random DA and No DA – and often by a large margin. For instance, GuDA yields returns 3x larger than the next best strategy (No DA) in antmaze-medium. GuDA with TD3+BC is also the only strategy that can solve antmaze-large with significance. While Random DA is often beneficial in maze2d and soccer-sim tasks, it often performs *worse* than No DA in other tasks. For instance, Random DA harms performance with all algorithms in antmaze-umaze, with BC and AWAC in antmaze-medium, and with BC and TD3+BC in parking. Since BC mimics the provided data, it is understandable that Random DA may harm performance with BC. However,



(a) (b)

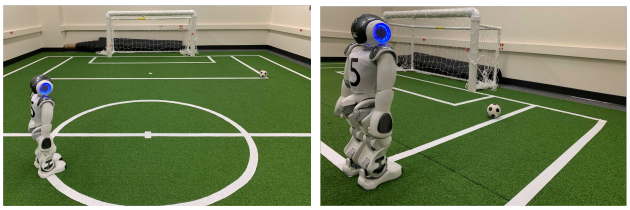
Fig. 4: Ablations studies on antmaze-medium with TD3+BC. (4a) 10k GuDA samples yields similar return to 1 million Random DA samples. (4b) GuDA outperforms Random DA with a larger initial dataset of 50k transitions.

since offline RL algorithms can in principle learn from suboptimal data, these findings emphasize the importance of generating expert augmented data even for offline RL.

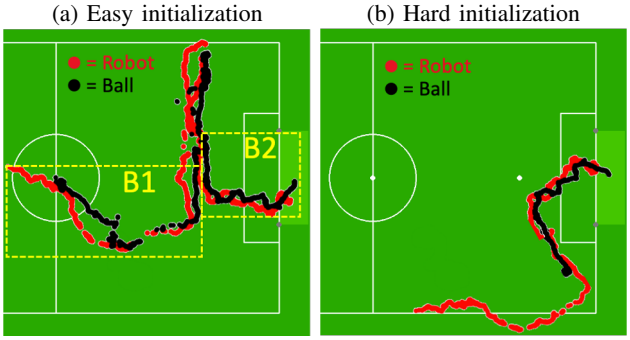
We additionally investigate the effect of (1) the number of augmentations we generate and (2) the size of our demonstration dataset. As shown in Fig. 4a, increasing the number of augmentations in general yields larger returns for both GuDA and Random DA, but GuDA can match Random DA's performance with far fewer augmentations. Moreover, Fig. 4b shows that GuDA outperforms Random DA if our initial dataset contains 50k transitions. Thus, GuDA can be beneficial even with abundant demonstration data.

C. Physical Experiments

We further evaluate GuDA in a physical robot soccer task in which a NAO V6 robot must score from the Easy and Hard initializations shown in Fig. 5a and 5b. The robot "kicks" the ball by simply walking into it. The ball's movements appear highly stochastic; they depend on how the robot's feet contact the ball and foot positions are not included as policy inputs. This stochasticity coupled with noisy vision-based state estimation makes this task notably difficult. We collect demonstrations using an expert policy pre-trained in a low-fidelity soccer simulator with simplified dynamics and perfect state estimation. Our demonstration dataset contains



(a) Easy initialization (b) Hard initialization



(c) Expert data (d) GuDA data

Fig. 5: (5a, 5b) Task initializations. (5c) Demonstration data with relevant behaviors B1 and B2. (5d) An illustration of GuDA data generated from the provided demonstration.

a single physical trajectory of the agent kicking the ball from the center of the field to the goal (Fig. 5c). This demonstration is highly suboptimal, as the robot fumbled the ball and had to take a circuitous route to the goal.

To apply GuDA, we first identify two task-relevant behaviors in our initial demonstration: (B1) the robot executing a tight turn to the ball, and (B2) the robot scoring with the ball away from the sideline (Fig. 5c). We use GuDA to generate augmented trajectories that trace out the path an expert might take to successfully score: we `Translate` and `Rotate` B1 to demonstrate the agent approaching the ball at a favorable angle, and we `Translate`, `Rotate`, and `Reflect` B2 to demonstrate the agent scoring with the ball away from the sideline. Because we use a physical demonstration, our augmented data accurately matches the task's true dynamics.

We generate 1 million augmented samples using GuDA and Random DA and train agents using IQL [72] for 1 million policy updates. We also compare agents to the expert demonstrator (Expert). Table II and Fig. 6 show the success rate and IQM time to score for each agent.[2] With the Easy initialization, GuDA scores faster and more frequently than Random DA and No DA. GuDA and expert policies have similar success rates, but GuDA scores significantly faster than the expert as well. We attribute this speedup to how the GuDA policy trained on augmented data that matches the physical world's dynamics whereas our expert policy trained in a low-fidelity simulator. With the Hard initialization, only the GuDA agent can consistently score. Random DA and No DA policies always kick the ball out of bounds. Even the expert policy almost always fails. Our results show that GuDA not only outperforms Random DA but also enables an agent to surpass its demonstrator in a difficult physical task with just a single suboptimal demonstration.

[2]We include videos of trained policies in our submission.

Initialization	GuDA	Expert	Random DA	No DA
Easy	8/10	9/10	4/10	4/10
Hard	7/10	2/10	0/10	0/10

TABLE II: Success rates for our physical robot soccer experiments. "Expert" denotes the policy we used to collect our initial demonstration. Green highlight indicates statistical significance according to a t-test at a 95% confidence level.

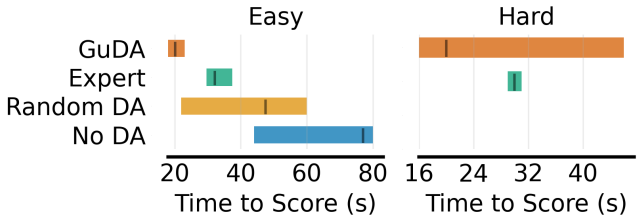


Fig. 6: IQM time to score with 95% bootstrapped confidence intervals. Lower times are better. GuDA's confidence interval in Hard is wide because of a single trial in which an usually hard kick moved the ball to the opposite end of the field.

VI. Conclusion

In this work, we introduced Guided Data Augmentation (GuDA), a human-guided data augmentation (DA) framework which generates expert-quality augmented data without the expense of real-world task interaction. In GuDA, a user imposes a series of simple rules on the DA process to automatically generate augmented samples that approximate expert behavior. GuDA can serve as a intuitive way to integrate human expertise into offline learning from demonstrations; instead of requiring that an expert demonstrate a near-optimal sequence of actions to solve a task, GuDA simply requires the user to understand what augmented data represents progress towards task completion. Empirically, we demonstrate that GuDA outperforms a widely applied random DA strategy and enables offline learning from a limited set of potentially suboptimal demonstrations. Furthermore, we show how GuDA yields an effective policy in a physical robot soccer task when given a single highly suboptimal trajectory. Our findings emphasize how a more intentional approach to DA can yield substantial performance gains.

The core limitation of GuDA is that it requires domain knowledge to specify sampling rules. Since the sampling rules required to generate expert augmented data are task dependent, GuDA must be implemented separately for each task. Nevertheless, these rules can be derived from basic intuitions on what task progress looks like and are simple to implement. While our empirical analysis focuses on behavior cloning and offline RL, GuDA can in principle be applied to other learning methods – both offline and online. In future work, we intend to study how GuDA interacts with other learning methods such as inverse RL and online RL. Furthermore, given the effectiveness of DA, we plan to conduct a broader analysis investigating the the most effective way to integrate augmented data into offline RL. Such an analysis would further strengthen the effectiveness of GuDA as well as other DA techniques.

References

[1] T. Apgar, P. Clary, K. Green, A. Fern, and J. W. Hurst, "Fast online trajectory optimization for the bipedal robot cassie." in *Robotics: Science and Systems*, vol. 101. Pittsburgh, Pennsylvania, USA, 2018, p. 14.

[2] G. Bledt, M. J. Powell, B. Katz, J. Di Carlo, P. M. Wensing, and S. Kim, "Mit cheetah 3: Design and control of a robust, dynamic quadruped robot," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 2245–2252.

[3] C. Gehring, S. Coros, M. Hutter, C. Dario Bellicoso, H. Heijnen, R. Diethelm, M. Bloesch, P. Fankhauser, J. Hwangbo, M. Hoepflinger, and R. Siegwart, "Practice makes perfect: An optimization-based approach to controlling agile motions for a quadruped robot," *IEEE Robotics Automation Magazine*, vol. 23, no. 1, pp. 34–43, 2016.

[4] T. Osa, J. Pajarinen, G. Neumann, J. A. Bagnell, P. Abbeel, J. Peters *et al.*, "An algorithmic perspective on imitation learning," *Foundations and Trends® in Robotics*, vol. 7, no. 1-2, pp. 1–179, 2018.

[5] A. Kumar, J. Fu, G. Tucker, and S. Levine, "Stabilizing off-policy q-learning via bootstrapping error reduction," in *Neural Information Processing Systems*, 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:173990380

[6] S. Fujimoto, D. Meger, and D. Precup, "Off-policy deep reinforcement learning without exploration," in *International conference on machine learning*. PMLR, 2019, pp. 2052–2062.

[7] A. Kumar, A. Zhou, G. Tucker, and S. Levine, "Conservative q-learning for offline reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 1179–1191, 2020.

[8] S. Fujimoto and S. S. Gu, "A minimalist approach to offline reinforcement learning," *Advances in neural information processing systems*, vol. 34, pp. 20 132–20 145, 2021.

[9] S. Levine, A. Kumar, G. Tucker, and J. Fu, "Offline reinforcement learning: Tutorial, review, and perspectives on open problems," *arXiv preprint arXiv:2005.01643*, 2020.

[10] N. Ratliff, J. A. Bagnell, and S. S. Srinivasa, "Imitation learning for locomotion and manipulation," in *2007 7th IEEE-RAS International Conference on Humanoid Robots*. IEEE, 2007, pp. 392–397.

[11] D. Kalashnikov, A. Irpan, P. Pastor, J. Ibarz, A. Herzog, E. Jang, D. Quillen, E. Holly, M. Kalakrishnan, V. Vanhoucke *et al.*, "Scalable deep reinforcement learning for vision-based robotic manipulation," in *Conference on Robot Learning*. PMLR, 2018, pp. 651–673.

[12] A. Mandlekar, D. Xu, J. Wong, S. Nasiriany, C. Wang, R. Kulkarni, L. Fei-Fei, S. Savarese, Y. Zhu, and R. Mart'in-Mart'in, "What matters in learning from offline human demonstrations for robot manipulation," in *Conference on Robot Learning*, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:236956615

[13] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang *et al.*, "End to end learning for self-driving cars," *arXiv preprint arXiv:1604.07316*, 2016.

[14] L. Sun, C. Peng, W. Zhan, and M. Tomizuka, "A fast integrated planning and control framework for autonomous driving via imitation learning," in *Dynamic Systems and Control Conference*, vol. 51913. American Society of Mechanical Engineers, 2018, p. V003T37A012.

[15] H. Ravichandar, A. S. Polydoros, S. Chernova, and A. Billard, "Recent advances in robot learning from demonstration," *Annual review of control, robotics, and autonomous systems*, vol. 3, pp. 297–330, 2020.

[16] S. Pitis, E. Creager, and A. Garg, "Counterfactual data augmentation using locally factored dynamics," *Advances in Neural Information Processing Systems*, vol. 33, pp. 3976–3990, 2020.

[17] S. Pitis, E. Creager, A. Mandlekar, and A. Garg, "Mocoda: Model-based counterfactual data augmentation," *Advances in Neural Information Processing Systems*, vol. 35, pp. 18 143–18 156, 2022.

[18] F. Abdolhosseini, H. Y. Ling, Z. Xie, X. B. Peng, and M. Van de Panne, "On learning symmetric locomotion," in *Motion, Interaction and Games*, 2019, pp. 1–10.

[19] S. K. Mikhail Pavlov and S. M. Plis, "Run, skeleton, run: skeletal model in a physics-based simulation," *AAAI Spring Symposium Series*, 2018.

[20] S. Sinha, A. Mandlekar, and A. Garg, "S4rl: Surprisingly simple self-supervision for offline reinforcement learning in robotics," in *Proceedings of the 5th Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, A. Faust, D. Hsu, and G. Neumann, Eds., vol. 164. PMLR, 08–11 Nov 2022, pp. 907–917. [Online]. Available: https://proceedings.mlr.press/v164/sinha22a.html

[21] H.-T. Joo, I.-C. Baek, and K.-J. Kim, "A swapping target q-value technique for data augmentation in offline reinforcement learning," *IEEE Access*, vol. 10, pp. 57 369–57 382, 2022.

[22] D. Cho, D. Shim, and H. J. Kim, "S2p: State-conditioned image synthesis for data augmentation in offline reinforcement learning," *Advances in Neural Information Processing Systems*, 2022.

[23] C. Lu, B. Huang, K. Wang, J. M. Hernández-Lobato, K. Zhang, and B. Schölkopf, "Sample-efficient reinforcement learning via counterfactual-based data augmentation," in *Offline Reinforcement Learning - Workshop at the 34th Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[24] C. A. Hepburn and G. Montana, "Model-based trajectory stitching for improved offline reinforcement learning," *arXiv preprint arXiv:2211.11603*, 2022.

[25] K. Wang, H. Zhao, X. Luo, K. Ren, W. Zhang, and D. Li, "Bootstrapped transformer for offline reinforcement learning," *Advances in Neural Information Processing Systems*, 2022.

[26] J. Han and J. Kim, "Selective data augmentation for improving the performance of offline reinforcement learning," in *2022 22nd International Conference on Control, Automation and Systems (ICCAS)*, 2022, pp. 222–226.

[27] S. Ross, G. Gordon, and D. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," in *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 2011, pp. 627–635.

[28] A. Kumar, J. Hong, A. Singh, and S. Levine, "Should I run offline reinforcement learning or behavioral cloning?" in *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. [Online]. Available: https://openreview.net/forum?id=AP1MKT37rJ

[29] V. G. Goecks, G. M. Gremillion, V. J. Lawhern, J. Valasek, and N. R. Waytowich, "Integrating behavior cloning and reinforcement learning for improved performance in dense and sparse reward environments," in *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*, ser. AAMAS '20. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, 2020, p. 465–473.

[30] M. Laskey, J. Lee, R. Fox, A. Dragan, and K. Goldberg, "Dart: Noise injection for robust imitation learning," in *Conference on robot learning*. PMLR, 2017, pp. 143–156.

[31] S. Arora and P. Doshi, "A survey of inverse reinforcement learning: Challenges, methods and progress," *Artificial Intelligence*, vol. 297, p. 103500, 2021.

[32] J. Ho and S. Ermon, "Generative adversarial imitation learning," *Advances in neural information processing systems*, vol. 29, 2016.

[33] C. Finn, S. Levine, and P. Abbeel, "Guided cost learning: Deep inverse optimal control via policy optimization," in *International conference on machine learning*. PMLR, 2016, pp. 49–58.

[34] A. Singh, L. Yang, K. Hartikainen, C. Finn, and S. Levine, "End-to-end robotic reinforcement learning without reward engineering," *ArXiv*, vol. abs/1904.07854, 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:119295213

[35] X. Zhang, L. Sun, Z. Kuang, and M. Tomizuka, "Learning variable impedance control via inverse reinforcement learning for force-related tasks," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 2225–2232, 2021.

[36] C. Gulcehre, Z. Wang, A. Novikov, T. Paine, S. Gómez, K. Zolna, R. Agarwal, J. S. Merel, D. J. Mankowitz, C. Paduraru *et al.*, "Rl unplugged: A suite of benchmarks for offline reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 7248–7259, 2020.

[37] S. K. S. Ghasemipour, D. Schuurmans, and S. S. Gu, "Emaq: Expected-max q-learning operator for simple yet effective offline and online rl," in *International Conference on Machine Learning*. PMLR, 2021, pp. 3682–3691.

[38] W. Zhou, S. Bajracharya, and D. Held, "Plas: Latent action space for offline reinforcement learning," in *Conference on Robot Learning*. PMLR, 2021, pp. 1719–1735.

[39] A. Nair, A. Gupta, M. Dalal, and S. Levine, "Awac: Accelerating online reinforcement learning with offline datasets," *arXiv preprint arXiv:2006.09359*, 2020.

[40] H. Xu, X. Zhan, J. Li, and H. Yin, "Offline reinforcement learning with soft behavior regularization," *arXiv preprint arXiv:2110.07395*, 2021.

[41] A. Fawzi, M. Balog, A. Huang, T. Hubert, B. Romera-Paredes, M. Barekatain, A. Novikov, F. J. R Ruiz, J. Schrittwieser, G. Swirszcz *et al.*, "Discovering faster matrix multiplication algorithms with reinforcement learning," *Nature*, vol. 610, no. 7930, pp. 47–53, 2022.

[42] A. George, A. Bartsch, and A. B. Farimani, "Minimizing human assistance: Augmenting a single demonstration for deep reinforcement learning," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 5027–5033.

[43] P. Mitrano and D. Berenson, "Data augmentation for manipulation," *arXiv preprint arXiv:2205.02886*, 2022.

[44] M. Laskin, K. Lee, A. Stooke, L. Pinto, P. Abbeel, and A. Srinivas, "Reinforcement learning with augmented data," *Advances in neural information processing systems*, vol. 33, pp. 19 884–19 895, 2020.

[45] L. Guan, M. Verma, S. Guo, R. Zhang, and S. Kambhampati, "Widening the pipeline in human-guided reinforcement learning with explanation and context-aware data augmentation," in *NeurIPS*, 2021.

[46] K. Wang, B. Kang, J. Shao, and J. Feng, "Improving generalization in reinforcement learning with mixture regularization," *Advances in Neural Information Processing Systems*, vol. 33, pp. 7968–7978, 2020.

[47] D. Yarats, R. Fergus, A. Lazaric, and L. Pinto, "Mastering visual continuous control: Improved data-augmented reinforcement learning," *arXiv preprint arXiv:2107.09645*, 2021.

[48] R. Raileanu, M. Goldstein, D. Yarats, I. Kostrikov, and R. Fergus, "Automatic data augmentation for generalization in reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 34, pp. 5402–5415, 2021.

[49] N. Hansen and X. Wang, "Generalization in reinforcement learning by soft data augmentation," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 13 611–13 617.

[50] N. Hansen, H. Su, and X. Wang, "Stabilizing deep q-learning with convnets and vision transformers under data augmentation," *Advances in Neural Information Processing Systems*, vol. 34, pp. 3680–3693, 2021.

[51] L. Guan, M. Verma, S. Guo, R. Zhang, and S. Kambhampati, "Widening the pipeline in human-guided reinforcement learning with explanation and context-aware data augmentation," in *Neural Information Processing Systems*, 2020. [Online]. Available: https://api.semanticscholar.org/CorpusID:238227077

[52] S. Sinha, A. Mandlekar, and A. Garg, "S4rl: Surprisingly simple self-supervision for offline reinforcement learning in robotics," in *Proceedings of the 5th Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, A. Faust, D. Hsu, and G. Neumann, Eds., vol. 164. PMLR, 08–11 Nov 2022, pp. 907–917.

[53] M. Weissenbacher, S. Sinha, A. Garg, and K. Yoshinobu, "Koopman q-learning: Offline reinforcement learning via symmetries of dynamics," in *International Conference on Machine Learning*. PMLR, 2022, pp. 23 645–23 667.

[54] Y.-L. Qiao, J. Liang, V. Koltun, and M. C. Lin, "Efficient differentiable simulation of articulated bodies," in *International Conference on Machine Learning*. PMLR, 2021, pp. 8661–8671.

[55] F. Sadeghi and S. Levine, "Cad2rl: Real single-image flight without a single real image," *arXiv preprint arXiv:1611.04201*, 2016.

[56] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," in *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2017, pp. 23–30.

[57] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Sim-to-real transfer of robotic control with dynamics randomization," in *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 3803–3810.

[58] M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, O. Pieter Abbeel, and W. Zaremba, "Hindsight experience replay," *Advances in neural information processing systems*, vol. 30, 2017.

[59] P. Rauber, A. Ummadisingu, F. Mutz, and J. Schmidhuber, "Hindsight policy gradients," *arXiv preprint arXiv:1711.06006*, 2017.

[60] M. Fang, C. Zhou, B. Shi, B. Gong, J. Xu, and T. Zhang, "Dher: Hindsight experience replay for dynamic goals," in *International Conference on Learning Representations*, 2018.

[61] M. Fang, T. Zhou, Y. Du, L. Han, and Z. Zhang, "Curriculum-guided hindsight experience replay," *Advances in neural information processing systems*, vol. 32, 2019.

[62] H. Liu, A. Trott, R. Socher, and C. Xiong, "Competitive experience replay," *arXiv preprint arXiv:1902.00528*, 2019.

[63] R. S. Sutton, "Integrated architectures for learning, planning, and reacting based on approximating dynamic programming," in *Machine learning proceedings 1990*. Elsevier, 1990, pp. 216–224.

[64] S. Gu, T. Lillicrap, I. Sutskever, and S. Levine, "Continuous deep q-learning with model-based acceleration," in *International conference on machine learning*. PMLR, 2016, pp. 2829–2838.

[65] A. Venkatraman, R. Capobianco, L. Pinto, M. Hebert, D. Nardi, and J. A. Bagnell, "Improved learning of dynamics models for control," in *International Symposium on Experimental Robotics*. Springer, 2016, pp. 703–713.

[66] S. Racanière, T. Weber, D. Reichert, L. Buesing, A. Guez, D. Jimenez Rezende, A. Puigdomènech Badia, O. Vinyals, N. Heess, Y. Li *et al.*, "Imagination-augmented agents for deep reinforcement learning," *Advances in neural information processing systems*, vol. 30, 2017.

[67] M. L. Puterman, *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

[68] E. van der Pol, D. Worrall, H. van Hoof, F. Oliehoek, and M. Welling, "Mdp homomorphic networks: Group symmetries in reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 4199–4210, 2020.

[69] J. Fu, A. Kumar, O. Nachum, G. Tucker, and S. Levine, "D4rl: Datasets for deep data-driven reinforcement learning," 2020.

[70] E. Leurent, "An environment for autonomous driving decision-making," https://github.com/eleurent/highway-env, 2018.

[71] R. Agarwal, M. Schwarzer, P. S. Castro, A. C. Courville, and M. Bellemare, "Deep reinforcement learning at the edge of the statistical precipice," *Advances in neural information processing systems*, vol. 34, pp. 29 304–29 320, 2021.

[72] I. Kostrikov, A. Nair, and S. Levine, "Offline reinforcement learning with implicit q-learning," *ArXiv*, vol. abs/2110.06169, 2021.