Flock Navigation by Coordinated Shepherds via Reinforcement Learning

Yazied Hasan[1], John E. G. Baxter[1], César A. S. Castillo[1,2], Elena Delgado[1], and Lydia Tapia[1]

[1] University of New Mexico, Albuquerque, NM 87131, USA
[2] Universidad de Ingeniería y Tecnología, Lima, Peru

Abstract. Shepherding is the problem of guiding a group of passive sheep agents (a flock) from some start position to a goal region by influencing the sheep motion with active guiding agents (the shepherds). Existing solutions are limited, as heuristic solutions are often designed and tuned for specific environments and flock dynamics, while learning solutions are often limited to a single shepherd and/or few sheep agents, fixed flock dynamics, discrete environments, or rely on tuning of heuristic solutions. In order to provide a more general shepherding solution, we contribute a mapping of this problem to a previously addressed planning problem, active agents protecting a moving payload from passive agents in a crowd. This inversion of an existing solution directly facilitates the creation of a deep reinforcement learning (deep RL) model that provides cooperation between multiple shepherd agents, shows robustness to changes in flock dynamics, and requires no predefined shepherding strategy. We experiment on the effect of varying the number of sheep agents and the number of shepherd agents to gauge the performance and scaling of each method. We also test our method's robustness to positional observation noise and changed flock dynamics both with and without re-training. The experiments show that our deep RL solution shepherds as well as tuned heuristic methods, often with a reduced path length of the shepherds. Our solution also exhibited robustness to environmental situations that were unseen during training and high adaptability with simple re-training.

Keywords: Shepherding, Motion Planning, Deep Reinforcement Learning

1 Introduction

Shepherding is a difficult planning problem where passive agents (a flock) are navigated via forces induced by one or more external agents (shepherds) [1–3]. This problem has been explored with single [2] and multiple [3] shepherd agents, environments with obstacles [4], and in discrete [5–7] and continuous [8, 9] state and action spaces. Besides the obvious contribution to agriculture [1], the shepherding problem has also been extensively researched for its practical

application in several other fields, such as security [10], crowd control [11] and environmental protection [12].

Many solutions have been proposed to solve the shepherding problem. Examples include mimicking real life shepherding behavior [13], heuristic rule-based algorithms [3, 14] and machine learning solutions [1, 4]. Heuristic rule-based methods are simple to implement and scale with flock size, but are sensitive to changes in the environment and agent dynamics. Learning-based methods on the other hand are more resilient to those changes [4]. However, existing learning solutions have several limitations including limited agent counts, predefined dynamic-specific strategies, discrete environments, and the need for precise observations and knowledge of all agent positions.

In our previous work, Payload Protection , we devised a framework using deep reinforcement learning (deep RL) to train a team of active agents to protect a moving object in a crowded environment from passive moving agents [15]. This solution provided scalability and utilized limited observations in the active agents. In this work, we adapt the protection framework to instead guide a flock of passive sheep agents towards a goal region. We show that by mapping the Markov Decision Process formulation (MDP) and adjusting the reward structure accordingly, the agents can learn the new task using the same network structure, environment, and action spaces used in previously.

We contribute an adaptation of an existing solution for active multi-agents influencing passive agents. We compare the performance of our method to two baseline heuristic methods and evaluate success, path lengths, flock spread, and completion time with varying shepherd and flock counts, as well as with changes in the dynamics and observations. Results demonstrate that our deep RL method successfully completes the task and is robust against changes in agent counts, environment noise, and changes in dynamics.

2 Related Work

Many solutions to the shepherding problem have explored rule- and heuristic-based behaviors for both single and multiple shepherd agents. Some methods constructed rule sets to mimic the behavior of real-life sheepdogs by directing a single shepherd to switch between predefined behaviors of driving and collecting [2, 14]. Most extensions of these behaviors to multiple shepherds direct groups of shepherds either explicitly or implicitly to form arcs and lines [3, 16, 17] or circles behind or around flocks [13, 18–20]. Nearly all of the rule-based methods rely on complete knowledge of the flock center of mass or the exact positions of all sheep agents. Our method and the solution from Lee and Kim [16] use local information gathered in a limited sensing range. However, the latter solution uses a predefined algorithm that relies on manual parameter tuning while our method does not rely on predefined shepherd behaviors or formations.

The shepherding problem has also been tackled with learning-based solutions. Some reinforcement learning solutions for the classic shepherding problem have been found, but are limited considering they use only one shepherd and

have small flock sizes ($n \leq 3$) [4, 21, 22]. Reinforcement learning has been shown to be capable of handling multiple agents [23, 24], and many learning solutions for the shepherding problem do use multiple shepherds. However, those methods often rely on given information like a base strategy for optimization or steering points for path guidance [8, 9, 25, 22]. Learning solutions for shepherding have been explored using discretized environments and action spaces [5–7, 21]. Non-traditional shepherding scenarios have also been explored, such as the use of adversarial sheep agents [26], or goals of capture/elimination of sheep rather than occupancy of a specific region [27, 28]. In contrast with previously presented learning methods, our work addresses the classic multi-agent shepherding problem for large flock sizes and goal regions in continuous space, while avoiding the use of predefined strategies for shepherd behavior.

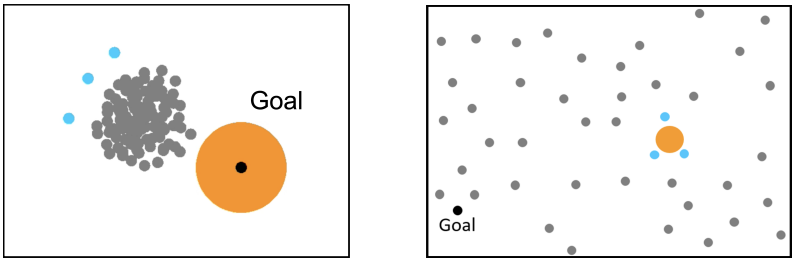
Most shepherding literature uses flock dynamics inspired by one of two models. One is the Reynolds 'boids' model, which uses three simple rules (separation, cohesion and velocity-matching) to produce life-like behavior in swarms [29]. Another is the Strömbom method, which models reactions of sheep agents to repulsive forces from shepherd agents [14]. This work focuses on the Strömbom dynamics model because of its direct relationship to the shepherding problem, but we also demonstrate that models can be trained to work on the Reynolds dynamics without changes in reward structure.

3 Problem Formulation

Our solution for the shepherding problem is based on our prior formulation of payload protection [15], which addresses the task of controlling homogeneous active agents (escorts, Fig. 1b, blue dots) to influence the movement of passive pedestrian agents (obstacles, Fig. 1b, gray dots) and prevent them from entering or colliding with the moving critical region (payload, Fig. 1b, filled orange circle). We represented the problem as a Multi-Agent Partially Observed Markov Decision Process (MA-POMDP), in the form of the tuple $(S, \mathcal{A}^N, \mathcal{O}, R, \mathcal{T}, \rho, \mathcal{N}, \mathcal{K}, \gamma)$, where S is the state space, $\mathcal{A}^N$ is the action space for the escorts, $\mathcal{O}$ is the set of observations, R is the reward structure, $\mathcal{T}$ is the transition function, ρ is the observation probability, $\mathcal{N}, \mathcal{K}$ are the sets of escorts and obstacles, and γ is the discount factor. The reward structure R engineered to solve the problem consisted of a penalty for obstacles colliding and breaching the payload region, and a reward for the payload reaching the goal.

The Shepherding Problem

Shepherding is the task of controlling one or more active agents (the shepherds) to guide a flock of homogeneous passive agents (the sheep) towards a critical region (the goal). The shepherd agents are controlled by some shared *policy*, and are represented by holonomic point particles (Fig. 1a, blue dots). The sheep are also represented by point particles (Fig. 1a, gray dots) and move according to flock dynamics that dictate how the herd reacts to other agents' positions



(a) Shepherding environment (b) Payload Protection environment

Fig. 1: The problem environments. The fundamental similarities in state, observation, and action spaces between the two environments motivates repurposing the solution from (b) to solve (a). (a) Shepherding environment with the sheep as gray dots, shepherds as blue dots, and the goal area as a filled orange circle. (b) Payload protection environment, with the moving obstacles as gray dots, the escorts as blue dots, and the payload as a filled orange circle.

and movements. We focus the majority of our work on dynamics that mimic real sheep [14], but the flock can use other dynamics models such as bird flocking [29]. The goal in this work is a circular goal region (Fig. 1a, orange circle).

We present the shepherding problem as a case of payload protection where the active agents *should* move the passive agents into a stationary payload region. By engineering a reward structure that motivates this inverted task, we can utilize the same deep RL framework from payload protection to approximate a policy for the shepherding problem. We map the set of escorts $\mathcal{N}$ to the set of shepherd agents $\mathcal{H}$, the obstacles $\mathcal{K}$ to the set of sheep agents $\mathcal{F}$, and the payload p to the goal g. The observation $\mathcal{O}$, state $\mathcal{S}$, action spaces $\mathcal{A}$, conditional observation probability ρ and discount factor γ stay the same across both problems. The state transition $\mathcal{T}$ then represents the dynamics of the shepherding problem, and the reward R is adjusted as previously mentioned. Thus the shepherding problem can also be represented as a MA-POMDP with tuple $\left(S, \mathcal{A}^{\mathcal{H}}, \mathcal{O}, R, \mathcal{T}, \rho, \mathcal{F}, \mathcal{H}, \gamma\right)$. At a given time, $s_g \in S_{\mathcal{G}}$, $s_f \in S_{\mathcal{F}}$, and $s_h \in S_{\mathcal{H}}$ are the states of the goal region, the f-th sheep agent, and the h-th shepherd agent. The state space S of the system is given by $S \equiv S_{g\,g\in\mathcal{G}} \times S_{f\,f\in\mathcal{F}} \times S_{h\,h\in\mathcal{H}}$.

At each step, for a given state $s \in S$, the shepherd agent $h \in \mathcal{H}$ receives an observation $o_h \in \mathcal{O}_h$, determined by the conditional observation probability $\rho\left(s, o_h\right) = P\left(o_h \mid s\right)$. The shepherd takes an action $a_h \in \mathcal{A}^h$ given by a policy, $\pi_\theta\left(o_h, a_h\right)$, with parameters θ. Given actions from all shepherd agents, a joint action $a_{h\,h\in\mathcal{H}} = a \in \mathcal{A}^{\mathcal{H}}$ is formed which induces transition in the environment according to the state transition function $\mathcal{T}\left(s, a, s'\right) = P\left(s' \mid s, a\right)$.

The observation of the h-th shepherd agent is a 1D simulated lidar with 512 rays equally distributed radially from the shepherd agent's center. Each ray has three channels: one for sheep agents, one for shepherd agents, and one for the goal region. Each ray channel returns the distance to the nearest circular surface

of an object of the channel's type along that ray, up to a maximum distance of $75m$. The range is chosen to be comparable to previous work [14, 13] with global vision for shepherds. If no object of the channel's type exists along that ray, a value of 0 is returned. To enable some inference of velocities and accelerations, the 2nd and 3rd derivatives of position, readings from the last three time steps are concatenated in a frame stack. Due to this setup, agents occlude other agents of the same type, but do not occlude those of different types.

For the action a_h in state s, the shepherd agent h receives a global reward $R(s, a_h)$. Each shepherd agent individually tries to maximize their expected cumulative reward, $E_{r\sim\pi}[R(\tau)]$, discounted by γ, where τ represents a sequence of states and actions of the shepherd agents following the policy π. Our representation turns the shepherding problem into the problem of finding parameters θ for policy π_θ that maps observations to shepherd agent actions which maximize reward. Policy π_θ is shared between all shepherd agents.

4 Method

We follow the learning setup from our previous work [15] and adapt it to the shepherding problem. We train multiple shepherd agents sharing one Generalized Advantage Estimation (GAE) [23] stochastic policy, similar to independent actor-critic with shared parameters [24, 23]. We share parameters because the shepherding agents are homogeneous. The policy is trained with Proximal Policy Optimization (PPO) [30]. Actor and critic are two separate, parallel networks receiving the same input. The input, as described in section 3, forms a $1,536 \times 1 \times 3$ size array (as shown in Fig. 2).

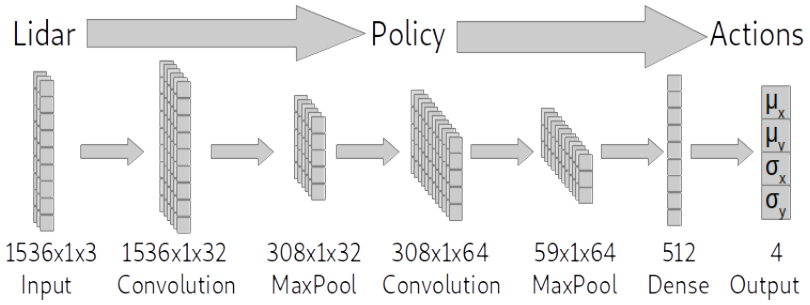


Fig. 2: Neural network architecture. The network takes in the sensor information from each type of sensed object: sheep agents, shepherd agents and goal region, and outputs a Gaussian distribution from which continuous actions are sampled. The network consists of alternating convolutional and max-Pooling layers followed by a single flattened dense layer. The mapping from payload protection to shepherding permits the use of identical architectures [15].

The output of the network is a set of continuous actions for each shepherd agent. Actions are represented in the neural network by a Gaussian distribution, $N\left([\mu v_x; \mu v_y], [\sigma v_x; \sigma v_y]\right)$, where μv_x and μv_y are means and σv_x and σv_y are standard deviations of the shepherd agents' horizontal and vertical speeds. The full network (Fig. 2) encodes a policy that maps input lidar information to output robot actions. The mapping is implemented through convolution layers (32 and 64 filters of size 1×10 and stride 1 with ReLU activation) and max pool layers (size 1×5 and stride 5). The output of the convolutional neural network is flattened and output to a dense hidden layer (size 512 with ReLU activation), which then returns the output action.

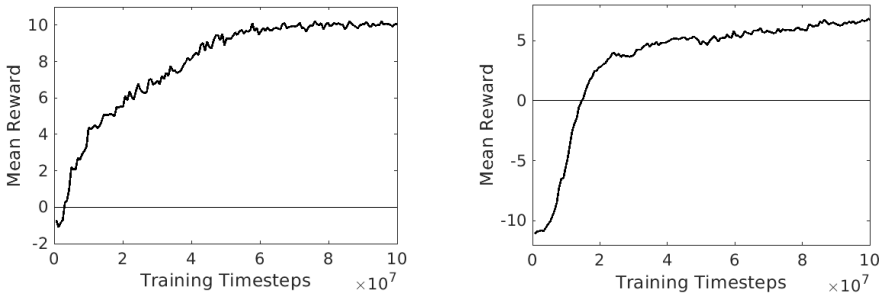
Episode initialization proceeds with the creation of a random environment. First, the goal region is placed at the center of a 50m by 50m square workspace. This position is purely for visualization , because the deep RL observations and comparison method calculations are all spatially relative. During training, the goal region has a radius uniformly sampled from $[4, 8]$m, while a fixed radius of 4m is used in evaluations. The flock is initialized in a random direction from the goal region with a center to center distance sampled uniformly from $[10, 20]$m. The shepherds are initialized around a point sampled uniformly in $[5, 10]$m from the flock center in a random direction. Individual sheep and shepherd agents are placed around their respective centers in random directions according to a Gaussian distribution with mean 0m and standard deviation 1m. In training, the flock size was 100, and between 1 and 6 shepherd agents were used to encourage generalization for different shepherd agent counts. Training episodes were 1000 timesteps long. Finally, a dynamics model and parameters are set.

Networks are trained and evaluated on two distinct models of flock dynamics. Table 1 describes the parameters of Strömbom and Reynolds flocking as detailed in [14, 29]. Both dynamics implement sheep-sheep repulsion, attraction to centers of mass, and repulsion from shepherds. The biggest difference between the two is what occurs when the shepherd is not near. In Strömbom dynamics, the flock stops moving except for occasional random movements, while in Reynolds dynamics there is local neighborhood velocity matching which causes the flock to constantly move. This difference, in addition to differences in weighting components of the force interactions, results in very different flock behaviors for which our networks learn very different policies.

Network training is driven by a reward function that rewards when sheep are in or close to the goal region. Specifically, we define two components of reward, *Occupancy Reward* and *Shaping Reward*. We define *Occupancy* as the number of sheep agents in the goal region normalized over the total flock size and episode timesteps. To devise *Occupancy Reward*, we multiply *Occupancy* by factor λ_o, here equal to 10. Using *Occupancy Reward* alone led to slow convergence, so we added a *Shaping Reward* to penalize the distance of sheep agents from the goal region. *Shaping Reward* is equal to one plus the distance from each sheep agent outside the goal region to the boundary of the goal region, normalized by the flock size and episode timesteps, then multiplied by λ_s, here equal to -0.1.Total Reward for each shepherd agent is then *Occupancy Reward* + *Shaping Reward*.

Strömbom Parameters	Description	Value
n	nearest neighbors	90
r_s	radius of sheep detection of shepherds	$\frac{65}{3}$m
r_a	agent to agent interaction distance	10m
ρ_a	repulsion from other sheep agents	2
c	attraction to sheep nearest neighbors	1.05
ρ_s	repulsion from the shepherd agents	1
h	inertia	0.5
e	angular noise	0.3
p	probability of moving while grazing	0.05
δ_a	sheep agent movement speed	$\frac{1}{3}$m/s
Reynolds Parameters	**Description**	**Value**
Neighborhood Radius	radius of sheep view	7.5m
Separation Factor	repulsion from other sheep agents	1.0
Velocity-Matching Factor	sheep heading and speed matching	7.5
Cohesion Factor	attraction to local center	1.0
Fear Factor	repulsion from shepherds	10.0
Sheep Speed	sheep agent movement speed	$\frac{1}{3}$m/s

Table 1: Dynamics parameters for Strömbom (top) and Reynolds (bottom) flock dynamics as applied to the sheep agents. Shepherd maximum speed is 1m/s.



(a) Strömbom flock dynamics [14]

(b) Reynolds flock dynamics [29]

Fig. 3: Deep RL policy training demonstrating neural network performance as experience is gained.

Training was performed for 100 million timesteps to learn policies that would navigate a flock to the goal region for a given flock dynamics type. Fig. 3 gives the mean cumulative reward plotted against the timesteps taken to train the models for (a) 'Strömbom' [14] and (b) 'Reynolds' [29] flock dynamics. While the flock dynamic model was changed for training, other environmental and training setup parameters remained fixed.

5 Experiments and Results

We assessed the performance of the trained deep RL models by evaluating them against two state-of-the-art shepherding methods. First we evaluate changes in performance as the numbers of shepherds and sheep increase. Next we measured method robustness by changing environments with elements not present during training. Finally, we evaluate the effect of retraining to accommodate different flock dynamics.Evaluation episodes were 1000 timesteps long.

We selected two state-of-the-art comparison methods that that can scale to multiple shepherds. The first comparison method, henceforth called 'Strömbom', is presented by Strömbom *et al.* [14] with enhancements by El Fiqi et al. [31]. The second comparison method, henceforth called 'Pierson', is presented by Pierson and Schwager [13]. Strömbom works by alternating between collecting behavior when the flock is separated and driving behavior when the flock is cohesive, and is representative for multi-modal driving point based method. We implemented enhancements from recent work [31] that allow multiple shepherds to use Strömbom through arc formations centered on the driving point, as well as avoid disrupting the flock by taking wide arc movements outside the effective range of repulsion. Pierson puts shepherds in an arc around the flock to control flock motion with unicycle-like dynamics, providing a non driving point based strategy. To generalize Pierson to one shepherd, the shepherd attempts to go to a place around the flock in the opposite direction of the desired heading of the flock at a distance proportional to flock spread.

We evaluate performance of our deep RL method and the two comparison methods on the shepherding problem using several metrics. The first is *completion time*, the number of timesteps it takes for the flock center of mass to enter the goal region. We also look at *path length*, the distance traveled by either the shepherd agent(s) or the flock center of mass. *Shepherd path length* is the distance traveled by the agent(s) throughout the episode normalized by the sum by the number of shepherds. *Flock path length* is calculated by summing the displacement of the flock center of mass over the episode. Another metric we use is *flock spread*, the standard deviation of the positions of the sheep agents in relation of the flock center of mass. The last metric we use is *flock fraction reaching goal*, which measures how much of the flock actually enters the goal region during an episode.

Scaling with Sheep and Shepherds

We measure the effect of the flock size on the performance of our learned shepherding policy and compare it to the baseline heuristic models. We changed the flock size from 10 to 100 in increments of 10 using Strombom flock dynamics, while keeping shepherds fixed at 3 agents. The results are shown in Fig. 4. From Fig. 4a, we see that Pierson achieves the fastest completion time, but Figs. 4b-d show that it does so at the cost of the path lengths and flock spread. This means that while Pierson reaches the goal in as little as two thirds of the time it takes deep RL to reach the goal, it does so by moving about four times more.

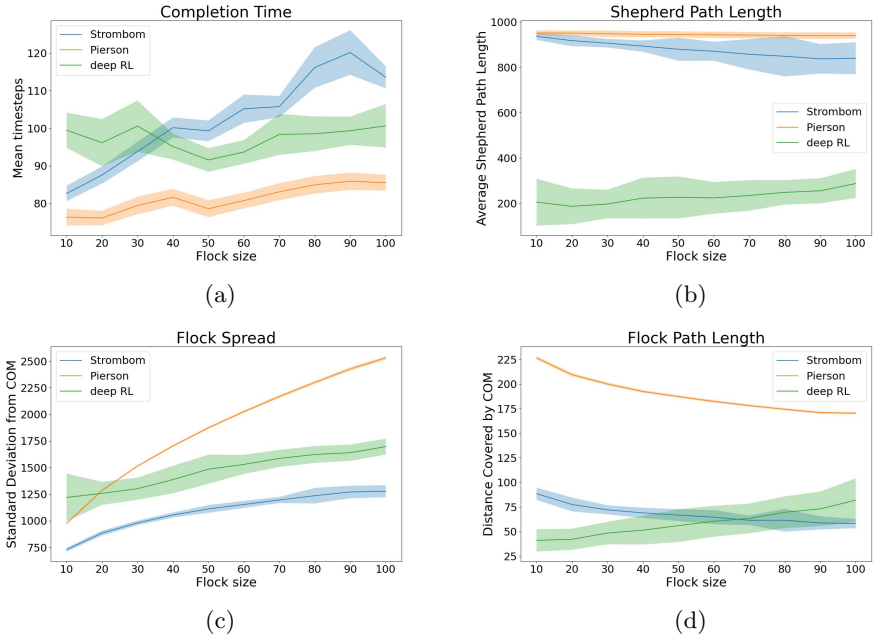


Fig. 4: Mean effects of increasing flock size on different metrics measured for each method. (a) Time taken for the flock center of mass (COM) to reach the goal. (b) Path length for the 3 shepherds. (c) Flock spread over all episodes. (d) Path length for the flock center of mass

Strömbom provides the lowest flock spread of the three methods (Fig. 4c), resulting in flocks with about 30% smaller spread than deep RL flocks. However, deep RL has the shortest paths for the shepherds (Fig. 4b) and the flock path length is comparable to that of Strömbom (Fig. 4d). It also shows similar flock spread scaling as Strömbom, increasing at similar rates as the flock size increases (Fig. 4c). Overall, each method stands out in a specific metric when scaling to flock size, but the deep RL method provides the shortest shepherd paths while being comparable to the state-of-the-art methods in their best metrics.

We also study the impact of increasing the number of shepherds on the performance of our deep RL shepherding policy compared to the baseline heuristic methods. For this evaluation, the number of shepherds was increased from 1 to 6 agents per episode while the flock size stayed fixed at 100 sheep with Strömbom flock dynamics. The results in Figs. 5a and 5c show that the deep RL method scales comparably to both heuristic methods in terms of completion times and flock spread as the number of shepherds increases. From Fig. 5a, we can see that all three methods benefit from increasing the shepherd counts, with diminishing returns after 3-4 shepherds. Figs. 5b and 5d show that while the deep RL performance worsens as the shepherd count increases, the method still outperforms

Pierson. We can see from Fig. 5b that the deep RL shepherd paths become longer with higher shepherd counts. That said, both Pierson and Strömbom take over double the path length of deep RL. Increased shepherd count impacts on the deep RL method are also seen with flock path length (Fig. 5d), where the deep RL method approaches about 80% of the Pierson path lengths. In Fig. 5c, deep RL flock spread is consistently less than Pierson but more than Strömbom. This means deep RL has more cohesive flocks than Pierson, but still has about 50% larger spread than Strömbom. While the deep RL path lengths increase with shepherd count, the completion time reduces, and the flock spread is consistent.

Additionally, we measure the fraction of the flock that reached the goal while varying both flock size and shepherd count. As results were highly consistent across changing flock sizes and shepherd counts, they are not shown in any figure. On average Pierson guides 100% of the flock to the goal, Strömbom guides 95.8% of the flock to the goal, and deep RL guides 97.8% of the flock to the goal.

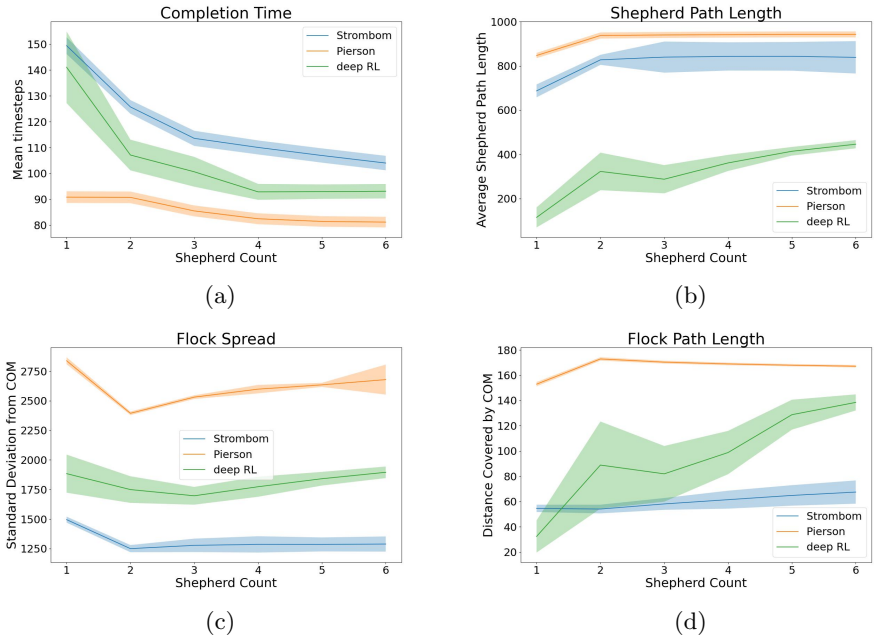
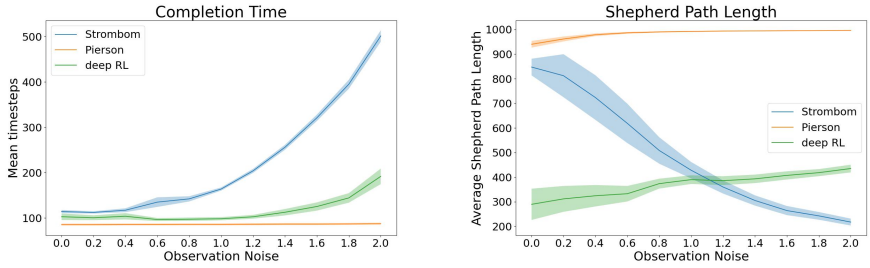


Fig. 5: Mean effects of increasing the number of shepherds on different metrics measured for each method. (a) Time taken for the flock center of mass (COM) to reach the goal. (b) Path length for the shepherds. (c) Flock spread over all episodes. (d) Path length for the flock center of mass. Flock size is 100.



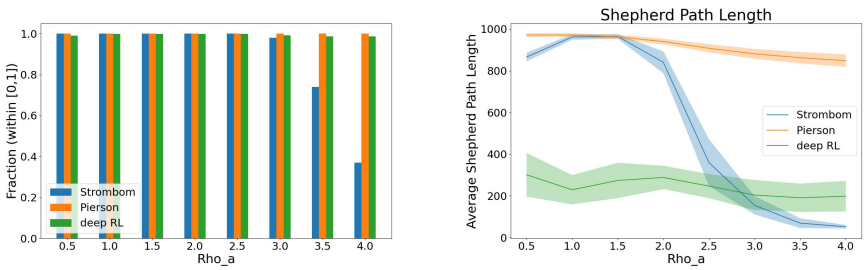
(a) (b)

Fig. 6: The effects of adding noise to the positional observation of the entities in the environment under Strömbom flock dynamics. (a) Time taken for the flock center of mass (COM) to reach the goal. (b) Average shepherd path length. Success rate is not shown, as all methods successfully completed the task.

Robustness to Observation and Changes in Dynamics

To showcase the deep RL method's robustness to uncertainty and noise in the environment, we evaluate the performance of the learned model against the heuristic baseline methods with the addition of noise applied to the entity positional information. In order to simulate observation noise, positions of the goal, shepherds, and sheep are all given Gaussian noise of zero mean with a standard deviation of 0 to 2m in each dimension. The new positions are used for the deep RL observations and the strategy for the heuristic methods. The noise does not affect the actual positions, metrics, or flock behavior. The evaluation episodes are performed with 3 shepherd agents and a flock size of 100 with Strömbom flock dynamics. The results are shown in Fig. 6. Results from Fig. 6a show that the Strömbom method takes longer time to complete the task as the observation noise increases, starting at standard deviation of 0.6m. On the other hand, deep RL completion time doesn't increase until 1.4m standard deviation. The Pierson method maintains constant completion time. Fig. 6b shows that the shepherd path length of the Pierson method is similar to previous experiments, consistently taking the most effort to perform the task. It takes twice as much energy as the deep RL and 5 times as much as Strömbom at high noise levels. Deep RL achieves lower path lengths than Strömbom until the observation noise reaches standard deviation 1m. Visual examination of episodes reveals that as the noise increases, the Strömbom shepherds are more likely to pause and stutter as they erroneously believe they are too close to the sheep, leading to lower path lengths. Overall, while Strömbom achieves shorter path lengths than deep RL, it does so at the expense of completion time. This indicates that deep RL is suitable for noisy environments when path lengths are of concern. Note that metrics were consistently high across the noise values tested and, as such, are not shown.

We also show the robustness of our method to variance in the dynamics of the flock. In this study, ρ_a, the repulsion factor in Strömbom flock dynamics, is



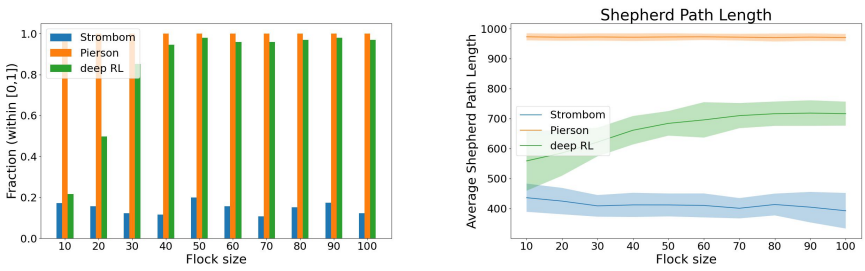
(a) (b)

Fig. 7: The effects of varying the flock cohesion parameter of Strömbom flock dynamics. (a) Fraction of sheep agents that reached the goal region averaged across all episodes. (b) Average shepherd path length.

varied to evaluate this sensitivity. The evaluation episodes are performed with 3 shepherd agents and a flock of size 100 Strömbom sheep. The value of ρ_a is modified from 1, ranging from 0.5 (higher cohesion for the flock) to 4.0 (lower cohesion of the flock). The results are shown in Fig. 7. Fig. 7a shows that the Pierson method maintains high success as ρ_a increases (cohesion decreases). The Strömbom method maintains high success until reaching ρ_a=3. Then, it starts to fall until reaching 40% success at ρ_a=4. The deep RL method, on the other hand, maintains high performance even as the value of ρ_a increases. Additionally, Fig. 7b shows that the shepherd path lengths decrease as the value of ρ_a increases. The decrease for Pierson and deep RL is small compared to the Strömbom path length decrease. The Strömbom decrease can be attributed to the stopping behavior seen when the shepherds get too close to the flock, since the flock spreads more with higher ρ_a. The Strömbom method manages lower path lengths than deep RL when ρ_a reaches 3, but that is also when it experiences a significant loss in success. It should be noted that the lower shepherd path lengths for Strömbom at $\rho_a = 0.5$ is due to the wide arc the shepherds take to avoid affecting the flock being smaller, making it more likely that the shepherds approach the flock and cause more pauses. The deep RL manages to maintain success as well as Pierson while also having lower path lengths. As opposed to the previous experiment, the deep RL method is overall more favorable than Strömbom for changes in the parameters of the dynamics.

Change in Flock Dynamics

We examine performance of the deep RL method and comparison methods when presented with a completely different set of dynamics, Reynolds (boids) flock model, which demonstrates less cohesion than Strömbom flock dynamics. The evaluation episodes have 3 shepherd agents with flock size of 10-100. The results from Fig. 8 confirm that the deep RL method can be trained on a different set of flock dynamics without adjusting the reward structure. Fig. 8a shows that

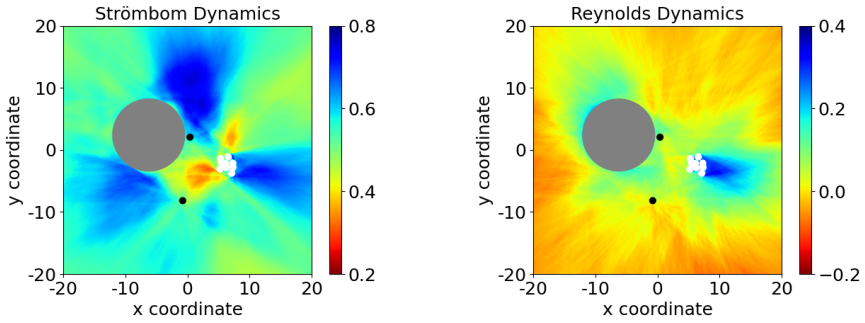


(a) (b)

Fig. 8: The effects of increasing flock size under Reynolds (boids) dynamics. (a) Fraction of sheep agents that reached the goal region averaged across all episodes. (b) Average shepherd path length.

Pierson achieves very high success with the new dynamics, despite not being tuned for it. Strömbom, on the other hand, consistently only has near 10% success as the flock size changes. The deep RL method achieves similar high performance when flock size is >30. Below a flock size of 30, deep RL has a lower success rate, potentially due to being trained on larger flock size of 100 sheep and not yet converging at the provided learning threshold. From Fig. 8b we can see that Strömbom maintains very low shepherd path lengths, less than 66% of deep RL. However, this can be attributed to the low success of the method with Reynolds dynamics which causes the flock to be much more dispersed. The deep RL method manages to scale well with flock size in terms of shepherd path lengths even when the flock has more complex dynamics. It is worth noting that this experiment, as opposed to previous ones, involved retraining the deep RL method so that the new flock dynamics could be observed. However, no other changes were done to the learning setup.

We compare the learned policy on the Strombom flock to that on the Reynolds flock using a value heatmap. The heatmap shows the estimated value of a shepherd agent in the presence of a given state containing the goal, two other shepherds, and 10 sheep. The results are shown in Fig. 9, where the value recorded at each X, Y coordinate represents the value estimated by the shepherd if placed in that position. We can see that both models prefer placing the third shepherd behind the flock, guiding it to the goal. Fig. 9a has large areas of high value. We attribute this to the Strombom flock clumping together tightly and being easier to guide, allowing a lenient strategy. On the other hand, Fig. 9b has high-value areas that are small. We attribute this to the Reynolds flock being more spread out and harder to guide, requiring a more stringent strategy. The heatmaps show that the models do not simply approximate similar strategies for both dynamics, but develop unique policies appropriate for each flock.



(a) (b)

Fig. 9: The value heatmap, for (a) agents trained on the Strombom flock and (b) agents trained on the Reynolds flock, depicting the estimated value of a third shepherd agent at the coordinate position. Higher values in blue indicate more preferable positions, and lower values in red indicate less preferable positions. The goal, shepherds, and sheep are represented by grey, black, and white circles, respectively.

6 Conclusion

In this work, we adapted a learning solution to the payload protection Problem to create a new deep RL solution for the shepherding problem. The model learned a policy for shepherds to guide a flock of sheep towards a goal area. The learned solution scales with flock size and shepherd count, and can be trained on different flock dynamics. The experimental results show that while heuristic methods provide solutions to many problem setups, they often do so at the expense of shepherd motion. Additionally, the deep RL method provides automatic coordination of shepherds, motion based on observations of sheep positions, and robustness/generality to several changes in the problem setup including scaling numbers of agents, positional noise, and changes in dynamics.

It is important to note the assumptions made in this work along with the subsequent limitations. First, this work focuses on the guiding and driving of the flock towards the goal area, rather than the collection of stragglers. As an assumption, the flock starts the episode in one cluster instead of spread about the environment. Another assumption we made is that, while the method does not assume perfect knowledge of flock agent positions and center of mass like other methods, different observation channel types do not occlude each other. That is, one sheep can occlude another sheep in the observation, but sheep do not occlude the goal from the shepherd's vision. This could be addressed with different inputs to the learning at a potential loss in solution quality due to the reduced positioning knowledge. Additionally, this work does not address the lack of completeness in shepherding solutions. Learning provides an approximate solution to this problem, so there is no guarantee the sheep agents will reach the goal region. However, this is the case for any learned solution.

Acknowledgement. The authors acknowledge Evan Carter of the Army Research Lab for helpful discussions. Tapia and Delgado are partially supported by the National Science Foundation under Grant Number IIS-1553266. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation. Hasan supported by the Saudi Arabian Cultural Mission to the United States. Computational resources primarily supported by the Air Force Research Laboratory under agreement number FA9453-18-2-0022. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon.

References

1. Schultz, A., Grefenstette, J., Adams, W.: Roboshepherd: Learning a complex behavior. In: Proc. Int. Symposium on Robot. and Autom. (1996)
2. Lien, J.M., Bayazit, O., Sowell, R., Rodriguez, S., Amato, N.: Shepherding behaviors. In: Proc. IEEE Int. Conf. Robot. Autom. (ICRA). vol. 4, pp. 4159–4164 Vol.4 (2004)
3. Lien, J.M., Rodriguez, S., Malric, J., Amato, N.: Shepherding behaviors with multiple shepherds. In: Proc. IEEE Int. Conf. Robot. Autom. (ICRA). pp. 3402–3407 (2005)
4. Zhi, J., Lien, J.M.: Learning to herd agents amongst obstacles: Training robust shepherding behaviors using deep reinforcement learning. Robot. and Automat. Lett. 6(2), 4163–4168 (2021)
5. Gadre, A.S.: Learning strategies in multi-agent systems-applications to the herding problem. Ph.D. thesis, Virginia Tech (2001)
6. Mahdavimoghaddam, M., Nikanjam, A., Abdoos, M.: Improved reinforcement learning in cooperative multi-agent environments using knowledge transfer. Computing Research Repository (CoRR) in arXiv (2022)
7. Go, C.K., Lao, B., Yoshimoto, J., Ikeda, K.: A reinforcement learning approach to the shepherding task using SARSA. In: Proc. 2016 Int. Joint Conf. on Neural Networks (IJCNN). pp. 3833–3836 (2016)
8. Brulé, J., Engel, K., Fung, N., Julien, I.: Evolving shepherding behavior with genetic programming algorithms. Computing Research Repository (CoRR) in arXiv (2016)
9. Özdemir, A., Gauci, M., Groß, R.: Shepherding with robots that do not compute. In: Proc. ECAL 2017, the Fourteenth European Conf. on Artificial Life. pp. 332–339. MIT Press (2017)
10. Shell, D., Mataric, M.: Directional audio beacon deployment: an assistive multi-robot application. In: Proc. IEEE Int. Conf. Robot. Autom. (ICRA). vol. 3, pp. 2588–2594 Vol.3 (2004)
11. Kirkland, J., Maciejewski, A.: A simulation of attempts to influence crowd dynamics. In: Proc. 2003 IEEE Int. Conf. on Systems, Man and Cybernetics. (Cat. No.03CH37483). vol. 5, pp. 4328–4333 vol.5 (2003)
12. Fingas, M.: The Basics of Oil Spill Cleanup. CRC Press/Taylor —& Francis, Boca Raton, FL (2013)
13. Pierson, A., Schwager, M.: Bio-inspired non-cooperative multi-robot herding. In: Proc. IEEE Int. Conf. Robot. Autom. (ICRA). pp. 1843–1849 (2015)
14. Strömbom, D., Mann, R.P., Wilson, A.M., Hailes, S., Morton, A.J., Sumpter, D.J.T., King, A.J.: Solving the shepherding problem: Heuristics for herding autonomous, interacting agents. J. R. Soc. Interface 11(20140719), 1–9 (2014)

15. Hasan, Y.A., Garg, A., Sugaya, S., Tapia, L.: Defensive escort teams for navigation in crowds via multi-agent deep reinforcement learning. Robot. and Automat. Lett. 5(4), 5645–5652 (2020)
16. Lee, W., Kim, D.: Autonomous shepherding behaviors of multiple target steering robots. Sensors 17(12) (2017)
17. Aiba, C., Fujioka, K.: A suggestion for effective shepherding models with two sheepdogs. In: Proc. Conf. IEEE Indust. Electronics Soc. (IECON). pp. 77–81 (2020)
18. Varava, A., Hang, K., Kragic, D., Pokorny, F.: Herding by caging: a topological approach towards guiding moving agents via mobile robots. In: Proc. Robotics: Sci. Sys. (RSS) (2017)
19. Song, H., Varava, A., Kravchenko, O., Kragic, D., Wang, M.Y., Pokorny, F.T., Hang, K.: Herding by caging: A formation-based motion planning framework for guiding mobile agents. Auton. Robots 45, 613–631 (2021)
20. Gade, S., Paranjape, A.A., Chung, S.J.: Robotic Herding Using Wavefront Algorithm: Performance and Stability, pp. 1–16. AIAA (2016)
21. Baumann, M., Buning, H.: Learning shepherding behavior. Ph.D. thesis, University of Paderborn (2016)
22. Nguyen, H.T., Nguyen, T.D., Garratt, M., Kasmarik, K., Anavatti, S., Barlow, M., Abbass, H.A.: A deep hierarchical reinforcement learner for aerial shepherding of ground swarms. In: Proc. Neural Information Processing: 26th Int. Conf., ICONIP 2019, Part I. p. 658–669 (2019)
23. Foerster, J.N., Farquhar, G., Afouras, T., Nardelli, N., Whiteson, S.: Counterfactual multi-agent policy gradients. In: Proc. AAAI Conf. on Artificial Intelligence. pp. 2974–2982 (Feb 2017)
24. K. Gupta, J., Egorov, M., Kochenderfer, M.: Cooperative multi-agent control using deep reinforcement learning. In: Proc. of Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS). pp. 66–83 (May 2017)
25. Nguyen, T., Liu, J., Nguyen, H., Kasmarik, K., Anavatti, S., Garratt, M., Abbass, H.: Perceptron-learning for scalable and transparent dynamic formation in swarm-on-swarm shepherding. In: Proc. 2020 Int. Joint Conf. on Neural Networks (IJCNN). pp. 1–8 (2020)
26. Potter, M.A., Meeden, L.A., Schultz, A.C.: Heterogeneity in the coevolved behaviors of mobile robots: The emergence of specialists. In: Proc. Int. Joint Conf. on Artificial Intelligence. vol. 17, pp. 1337–1343. Citeseer (2001)
27. Kowalczuk, Z., Jedruch, W., Szymański, K.: The use of an autoencoder in the problem of shepherding. In: Proc. 2018 23rd Int. Conf. on Methods Models in Automation Robotics (MMAR). pp. 947–952 (2018)
28. Georgiev, M., Tanev, I., Shimohara, K., Ray, T.: Evolution, robustness and generality of a team of simple agents with asymmetric morphology in predator-prey pursuit problem. Information 10(2) (2019)
29. Reynolds, C.W.: Flocks, herds and schools: A distributed behavioral model. In: Proc. ACM SIGGRAPH. p. 25–34 (1987)
30. Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O.: Proximal policy optimization algorithms. Computing Research Repository (CoRR) in arXiv (2017)
31. El-Fiqi, H., Campbell, B., Elsayed, S., Perry, A., Singh, H.K., Hunjet, R., Abbass, H.A.: The limits of reactive shepherding approaches for swarm guidance. IEEE Access 8, 214658–214671 (2020)