Behavior-based Model for Sheepdog and Sheep Herding Using Social Force

Noah James and John Ortiz

Introduction

Recent advances in sheep-sheepdog agent modeling has mainly focused on reinforcement learning due to its powerful results. However, it is important to continue to test behavior-based models that offer direct insight into the decision making process in biological agents, as behavior-based approaches are inherently easier to interpret than black box neural networks. Because a sheep herd is analogous to a swarm, advances in this area of research can be extrapolated to various complex engineering tasks, such as swarm robotics or unmanned vehicle navigation [1, 2]. In this paper, we examine and test the work of *Cai et al.* [3], which outlines a modern behavior-based approach to modeling sheepdog herding.

Although many sheep herding models use flocking as the underlying mechanism for guiding sheep behavior [4], *Cai et al.* chose to incorporate the social force model into their study [5]. The social force is simply the summation of attractive and repulsive forces exerted by neighboring agents onto a given agent, and the distance between the agent and each of its neighbors determines whether the force is attractive or repulsive. This is divided into four circular zones, where the innermost zone (closest to the agent) is repulsive, the next zone is neutral with no force being applied, and the third zone is attractive. Outside of the third zone, the agent no longer has vision and thus there are no forces applied from agents here.

In real-world examples of sheepdog behavior, the sheepdog follows a sweeping back-and-forth pattern to prevent sheep from breaking off from the herd and to effectively average the propagation of the herd towards the destination. This is replicated in the algorithms comprising this model, and we show through our experiments that it is also effective in successfully moving the herd to the destination.

Related Work/Background

Agent Based Modeling

The field of sheep herding using agent based models has seen significant advancements in recent years, with researchers exploring various techniques to improve the performance of autonomous herding agents. Among these techniques, Reinforcement Learning has emerged as the most popular approach, owing to its ability to learn optimal policies through interactions with the environment. *Hussein et al.* contributed largely to these RL models as their paper explores it using curriculum based reinforcement learning on the sheepdog [6].

Behavior Based Modeling

The model chosen to explore is a behavior model based on a social force model proposed in *Mehran et al.* paper [5] used to describe abnormal crowd behavior.

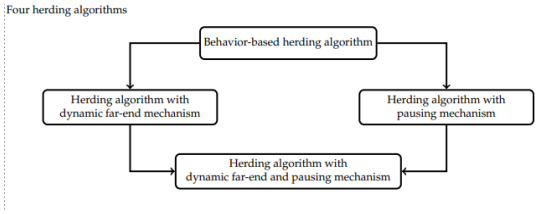
Methodology

Based on the original model design, we recreated the experimental space described in the *Cai et al.* paper [3]. This means defining a target area (with a soft boundary[1]), populating a sheep herd in a defined area, and having the sheepdog start initially away from the target area and sheep herd

The novelty of the model is from the algorithms that determine the movement of the sheepdog. The formulated four algorithms that will be explored to model the sheepdogs behavior are all derived from the base

[1] A soft boundary means that agents(sheep) can exit the target area once pushed in already.

algorithm with the Algorithm 4 being the combination of the alterations(depicted below).



The base algorithm should result with side to side motion of the sheepdog toward the target area with the sheep herd in between.

The sheep dog does this by selecting critical sheep based on some selection criteria. There are anchor sheep given by these equations:

$$\mathbb{V}^{la}(k) = \underset{i \in \mathbb{V}_q^l(k)}{\arg\min}\, F(i),\ \mathbb{V}^{ra}(k) = \underset{i \in \mathbb{V}_q^r(k)}{\arg\min}\, F(i)$$

$$\mathcal{P}^{la}(k) = \underset{i \in \mathbb{V}^{la}(k)}{\arg\min}\, G(p_i(k)),\ \mathcal{P}^{ra}(k) = \underset{i \in \mathbb{V}^{ra}(k)}{\arg\min}\, G(p_i(k))$$

The other two sheep are front sheep given by

$$\mathbb{V}^{lf}(k) = \underset{i \in \underline{N}}{\arg\max}\, H^l(i),\ \mathbb{V}^{rf}(k) = \underset{i \in \underline{N}}{\arg\max}\, H^r(i)$$

$$\mathcal{P}^{lf}(k) = \underset{i \in \mathbb{V}^{lf}(k)}{\arg\max}\, G(p_i(k)),\ \mathcal{P}^{rf}(k) = \underset{i \in \mathbb{V}^{rf}(k)}{\arg\max}\, G(p_i(k))$$

Where these functions used to define the above anchor sheep:

$$H^l(i) = \langle p_i^q(k), D^{cdl}(k) \rangle,\ H^r(i) = \langle p_i^q(k), D^{cdr}(k) \rangle$$
$$F(i) = \langle p_i^q(k), D^{qc}(k) \rangle,\ G(p_i(k)) = \|p_i^q(k)\|$$

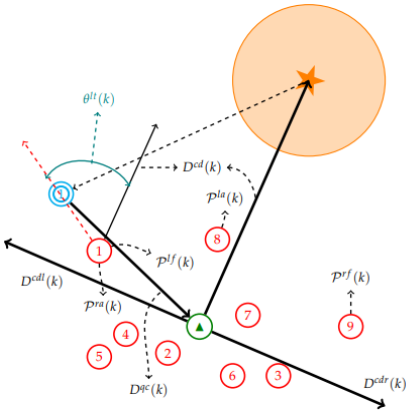
Where sheep partitions V(k) are essentially defining sheep that are on the left and right of the unit vector created from sheepdog to the center of the visible sheep given by the following:

$$\mathbb{V}_q^l(k) = \{i | p_i^q(k) \in \mathbb{S}_l(D^{qc}(k))\},\ \mathbb{V}_q^r(k) = \{i | p_i^q(k) \in \mathbb{S}_r(D^{qc}(k))\}.$$

And where the unit vectors defining direction used are defined below:

$$D^{cd}(k) = \mathbf{o}(p_d - p^c(k))$$
$$D^{cdl}(k) = \mathbf{R}\left(\frac{\pi}{2}\right) D^{cd}(k)$$
$$D^{cdr}(k) = \mathbf{R}\left(-\frac{\pi}{2}\right) D^{cd}(k)$$
$$D^{qc}(k) = \mathbf{o}(p^c(k) - q(k)).$$

Where disregarding all the nomenclature essentially means the two front sheep are in respect from the centroid of the sheep to the center of the target area where both sheep fall on the left and ride side of the unit vector created between them. The two anchor sheep are in respect to the sheep dog and centroid of the sheep being the two closest to the sheepdog and falling on the left and right of the unit vector between. This is depicted in the below figure courtesy *Cai et al [3]*.



Going back to the sheepdog algorithm design with the base of the algorithm, slight modifications were made to improve model performance. First being by instead of having the sheepdog use the center of the target area as the reference point for the goal instead opting to give the far end of the target area from the location of the sheep dog. This ends up pushing sheep further into the target area which would help the success rate of the model since the target area is treated with a soft boundary. Looking at Algorithm 4, the highlighted blue is where the change from center of the target area was to the dynamic far end. The addition of pausing is based off of two parameters the back declination angle :

$$\theta^b(k) = \Theta(D^{cd}(k), D^{qc}(k)). \qquad \Theta(x,y) = \arccos\left(\frac{\langle x,y \rangle}{||x|| \cdot ||y||}\right)$$

And the approaching ratio

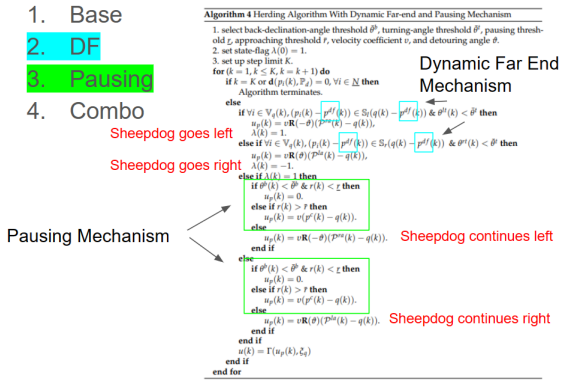
$$r(k) = \frac{\|q(k) - p^c(k)\|}{\max_{i \in \mathbb{V}_q(k)} \{\|p_i(k) - p^c(k)\|\}}.$$

Essentially, the back declination angle (θ^b) defines the angle between the unit vectors of the sheepdog's location minus the center of the sheep herd with the sheepdog's

location minus the center of the destination area. Hence if the back declination angle is small then the sheepdog is close to the center of the sheep herd in respect to the x direction not the y. The approaching ratios can be thought to be:

$$\frac{\text{distance(sheepDog to centroid of sheep)}}{\text{Furthest distance(sheepDog to any sheep)}}$$

The approaching ratio gets small as the sheep are more spread out and will become large when the sheep become close together. Knowing both these parameters we can say pausing should occur when back declination is small and the approaching ratio is small this should allow for the sheep herd to stay closer together without splitting into multiple herds and allow the social force imposed on the sheep by the sheepdog to propagate through the sheep herd. The highlighted green portions of Algorithm 4 are the additions made to incorporate pausing.

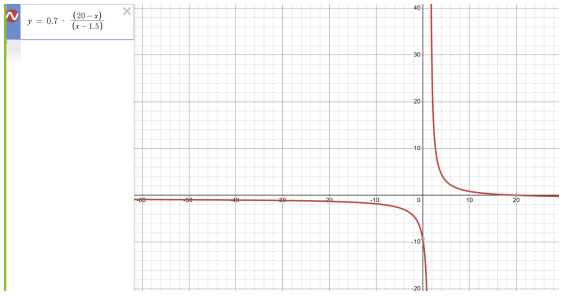


During the design and implementation of their [3] algorithms we noted some drawbacks. The original paper design makes the model not robust to certain conditions. It does not have a multi-step process for searching for the sheep herd when far away. As the assumption is that the sheep herd has already been located. In terms of comprehension the original paper had too much nomenclature trying to define specifics when it should have had less variables to make understanding the equations and logic easier to follow. As even with a lot of nomenclature the paper fails to define some important details including units for the size of the time steps,the size of the space, and measurements of distance (between agents). Hence for the latter details we made logical assumptions to fill in the gaps and will attempt to explain with as little nomenclature use as possible.[2] Another notable drawback noted while designing the model and algorithms was that the equation for selecting critical sheep was incomprehensible as it describes taking an argmax of an argmax which seems redundant and unnecessary.

The last drawback of the design appeared during testing when we noticed an issue with repulsion force, which is defined by:

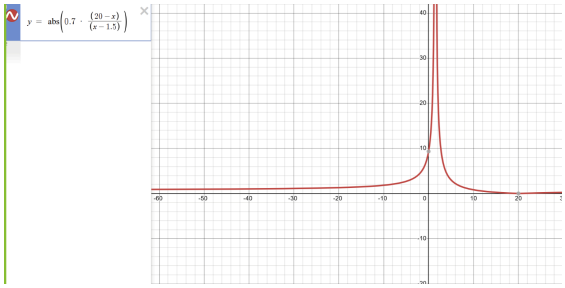
$$\varphi(x) = \alpha\left(\frac{\rho_{vp} - x}{x - \rho_x}\right)$$

As a sheep becomes closer to a sheepdog the original equation would act as the following graph depicts. This becomes an issue in a simulation which uses discrete time steps, since the dog, which is not affected by this force, can move past the minimal safety distance during a position update and flip the force to be negative. As a result, the sheep would be suddenly be attracted to the dog once it crosses this threshold.



To combat this error, we implemented an absolute value around the original function resulting in the following graph, which ensured that attraction between the sheep and sheepdog could not occur. If we did not change this repulsion function, this would inevitably become an issue as the sheep have a lower max speed than the sheepdog, meaning that it is guaranteed that the sheepdog can catch up to any particular sheep as long as it is moving towards it.

[2] All nomenclature defined in *Cai et al.* paper can be found in the appendix



Experimental Design

Procedure

For the experimental design, we utilized python and a library for simulation called AgentPy. After defining the four algorithms for the sheepdog, we ran 1000 trials (simulations) for each with a static target area and randomly generated sheep positions (relatively close together). Algorithm 1 is the base algorithm that locates the critical sheep and sweeps around them according to the social force. Algorithm 2 is the same as Algorithm 1 except it uses the dynamic far-end as opposed to the center of the destination area. Algorithm 3 implements the pausing mechanism without dynamic far-end, and Algorithm 4 implements both dynamic far-end and pausing. We also give an example of what performance with a single obstacle might look like in our simulation visualizations.

Agent Setup

There will be 24 sheep generated and 1 sheep dog generated per simulation. Each simulation will be defined on a 100 by 100 grid running simulations at 30 frames per second.

Statistics

For exploring the success of each algorithm we tracked the highest percentage of sheep herded, time it took to successfully complete, and success rate.

Results

The results of our experiments somewhat supported the findings in *Cai et al.*, but there were some discrepancies depending on the metric. Figure 1 shows the mean highest percentage achieved by each algorithm over 1000 trials. Since many of the trials were successes, Figure 1 effectively showcases the consistency of each algorithm, as any drastic failures significantly lowered the average percentage herded. This was the case for Algorithm 3 and Algorithm 4, which introduced the pausing mechanism and occasionally had failures with as low as 5% sheep herded, whereas Algorithm 1 and Algorithm 2 consistently succeeded or only missed 1 sheep during their simulations. Curiously enough, adding dynamic far-end in Algorithms 2 and 4 resulted in exactly a 0.5% increase in the mean percentage from Algorithms 1 and 3, respectively. This seems to imply that dynamic far-end is a consistent improvement upon methods that simply track the destination center.

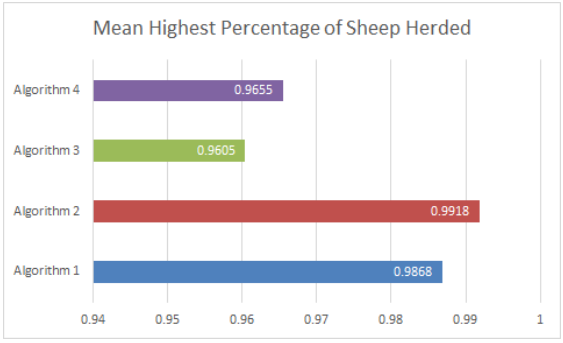


Figure 1: Highest percentage of sheep herded into the destination within 2500 timesteps, averaged over 1000 trials.

For success rate, we begin to see the expected results with Algorithms 3 and 4 performing better in Figure 2. The discrepancy between the data in Figure 1 and 2 is due to Algorithms 3 and 4 having more successful trials but also a few catastrophic failures. Algorithms 1 and 2, conversely, averaged far better but had trouble herding all 24 sheep into the destination area.

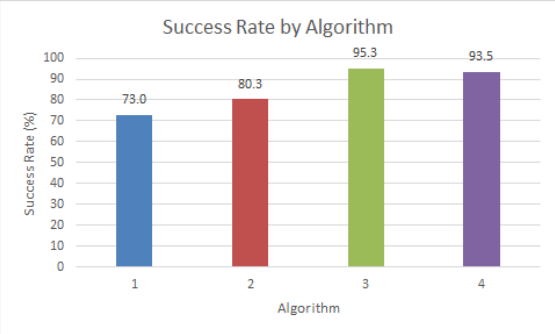


Figure 2: Success rate for each algorithm, where success is counted if all sheep are herded into the destination area at any point in 2500 time steps.

Finally, we recorded the number of time steps taken to successfully herd all of the sheep and displayed the results in Figure 3. The sheepdog would herd the fastest under Algorithm 4 and the slowest under Algorithm 1. This trend is understandable since Algorithms 3 and 4 introduce the pausing mechanism, which allows the sheep herd to regroup without the dog performing extra sweeps to catch escaping sheep.

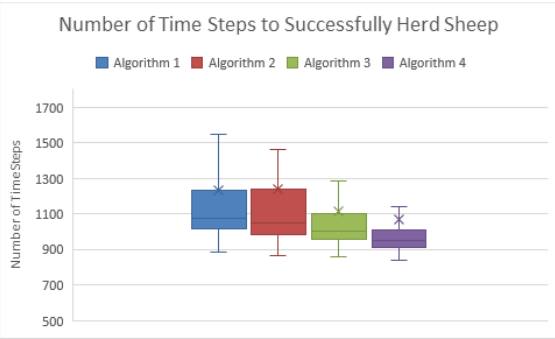


Figure 3: Number of time steps taken by the sheepdog to herd the sheep (if the trial was successful). The maximum number of time steps was 2500 for all trials.

Discussion

Findings

From our visualized simulations, we can see that these algorithms function as expected, with the sheepdog performing the back-and-forth sweeping motion to herd the sheep agents. With the addition of a dynamic far-end, the dog is more efficient at pushing the sheep herd into the soft destination. The data overall supports the results of *Cai et al.*, with success rate generally increasing as the algorithms increase in complexity. An interesting result of our experiments is that Algorithm 3 did outperform Algorithm 4 in success rate despite dynamic far-end effectively giving more space for the sheepdog to herd the sheep into the destination area. We suspect that this is a byproduct of whatever is causing seemingly random failures in the pausing mechanism trials. Since the success rates are quite similar, we can assume that Algorithms 3 and 4 are approximately the same in terms of accuracy.

Conclusion

Although our results for this set of algorithms are promising and serve to support the work of Cai et al., there are several aspects of the model that could be improved in future works. First and foremost, the model does not have considerations for obstacles, and if the agents did tend to avoid obstacles, the sheepdog is not structured to handle deviations in the sheep herd caused by them. The model also falters as the number of sheep in the simulation increases. This is a natural occurrence in real-world situations as well, since usually more than one sheepdog is needed to herd large sheep populations. This therefore motivates further research into developing a multi-sheepdog model for large herds. Aside from this, the model is also only defined under certain initial conditions (i.e. when the dog is already within vision of the sheep herd). To make it more robust, it would need an algorithm to handle herds of sheep that are far apart or distant from the sheepdog. We have listed several criticisms of [3] as well, such as the sheep-sheepdog repulsion function and the neglect to consider units in the simulation. All the above serve as good starting points for improvement in the field of behavioral sheepdog agent modeling.

Implementation and Simulated Results

Our implementation of this model can be found on GitHub, and our simulation visualizations can be found in this Google Drive folder.

Project Contributions

Initially for development of the model, Noah worked on the Social Force Model for the sheep, and John worked on implementing visualizations of the model simulations.. After that initial step, we combined our code and built the sheepdog agent and interfaced it with the existing sheep agent class. Then, Noah constructed the experimental setup for testing the model as John worked on developing obstacles to interact with. We ran into many issues in developing the code and thought it best to work on it together for the duration of the project (pair programming).

Appendix

Nomenclature from *Cai et al.* :

Nomenclature

$p_i(k)$	the position of the ith sheep at step k
$v_i(k)$	the velocity of the ith sheep at step k
$q(k)$	the position of the sheepdog at step k
$u(k)$	the velocity of the sheepdog at step k
T	the sampling period
$p_{ij}(k)$	the relative position of the jth sheep towards the ith sheep
$p_i^q(k)$	the relative position of the sheepdog towards the ith sheep
ρ_{vp}	perceptive radius of the sheep
ρ_{vq}	perceptive radius of the sheepdog
$\mathbb{V}_{pi}(k)$	the label set of all the sheep visible to sheep i at step k
$\mathbb{V}_q(k)$	the label set of all the sheep visible to the sheepdog at step k
$p^c(k)$	the center of the visible sheep set at step k
$v_{pi}(k)$	the effect from other sheep to the ith sheep at step k
$v_{qi}(k)$	the effect from the sheepdog to the ith sheep at step k
$n_i(k)$	noise
ξ_p	the upper limit for the speed of sheep
$\mathbb{P}_p(k)$	the sheep herd polygon at step k
$\mathbb{P}_d$	destination area
p_d	the center of $\mathbb{P}_d$
ρ_d	the radius of the destination area
$D^{cd}(k)$	the direction of sheep herd towards destination area at step k
$D^{cdl}(k)$	the left vertical of $D^{cd}(k)$
$D^{cdr}(k)$	the right vertical of $D^{cd}(k)$
$D^{qc}(k)$	the direction of the sheepdog towards the $p^c(k)$
$\mathcal{P}^{la}(k)$	the left anchor sheep at step k
$\mathcal{P}^{ra}(k)$	the right anchor sheep at step k
$\mathcal{P}^{lf}(k)$	the left front sheep at step k
$\mathcal{P}^{rf}(k)$	the right front sheep at step k
$\theta^{lt}(k)$	left turning angle at step k
$\theta^{rt}(k)$	right turning angle at step k
$\lambda(k)$	state flag at step k
ξ_q	the upper limit of sheepdog's speed
$p^{df}(k)$	the dynamic far-end at step k
$\theta^b(k)$	back declination angle of sheepdog relative to sheep herd at step k
$r(k)$	the approaching ratio at step k
ρ_ε	the allowable minimal distance between the sheep and the sheepdog
ρ_s	the minimal sheep-to-sheep safety distance
ρ_r	the maximal action distance of the repulsive effect between two sheep
ρ_g	the minimal action distance of the attractive effect between two sheep
α	the intensity coefficient of sheepdog-to-sheep repulsive force
β	the intensity coefficient of sheep-to-sheep repulsive force
γ	the intensity coefficient of sheep-to-sheep attractive force
θ^b	back-declination-angle threshold
θ^t	turning-angle threshold
$\underline{r}$	pausing threshold
$\bar{r}$	approaching threshold
v	velocity coefficient
θ	detouring angle
K	step limit
Ξ	success rate
κ	completion step
Υ	control cost

References

[1] Wang, X.; Yadav, V.; Balakrishnan, S. Cooperative UAV formation flying with obstacle/collision avoidance. IEEE Trans. Control. Syst. Technol. 2007, 15, 672–679

[2] Bayındır, L. A review of swarm robotics tasks. Neurocomputing 2016, 172, 292–321

[3] Cai, He & He, Yaqi & Wu, Jinye & Gao, Huanli. (2023). Behavior-Based Herding Algorithm for Social Force Model Based Sheep Herd. Electronics. 12. 285. 10.3390/electronics12020285.

[4] C. W. Reynolds, "Flocks, Herds and Schools: A Distributed Behavioral Model," in Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques, 1987, pp. 25–34. doi: 10.1145/37401.37406.

[5] Mehran, R.; Oyama, A.; Shah, M. Abnormal crowd behavior detection using social force model. In Proceedings of the 2009 IEEEConference on Computer Vision and Pattern Recognition, IEEE, Miami, FL, USA, 20–25 June 2009; pp. 935–942

[6] Hussein, Aya & Petraki, Eleni & Elsawah, Sondoss & Abbass, Hussein. (2022). Autonomous Swarm Shepherding Using Curriculum-Based Reinforcement Learning