FRIteza

Collective Behaviour Optimal Shepherding

Franziska Weber, Franz Muszarsky, and Kimberley Frings

Second Report

December 17, 2023

Iztok Lebar Bajec | izredni professor | mentor

In our initial report, we tackled the important sections of our chosen paper *Optimal Shepherding and Transport of a Flock*[1] and mainly concentrated on explaining how to run our model. For the second report, we build upon our previous work and will provide an overview of the current status of our project. We give a comprehensive description of our model. Additionally, we explore the implementation of two extensions to this model: a surrounding fence and the introduction of a second shepherd.

Methods

Model description. The herd consists of N agents which move in a two-dimensional open field. The behaviour of the agents is based on Reynolds' boids model [2]. To be more precise, the movement of each agent depends on three agent-agent interactions, namely local alignment, repulsion and weak attraction to the herd center, and on the repulsion from the shepherd.

This leads to the following velocity field of an agent in the herd, where α, β, γ and δ are weights:

$$\boldsymbol{v}^{\text{net}} = \alpha \boldsymbol{v}_{a-a}^{\text{alignment}} + \beta \boldsymbol{v}_{a-a}^{\text{repulsion}} + \gamma \boldsymbol{v}_{a-a}^{\text{attraction}} + \delta \boldsymbol{v}_{a-s}^{\text{repulsion}} \quad [1]$$

Local alignment means that agents which are close to each other align their velocity vectors. We use the formulation from the Vicsek model [3]: At each timestep, the orientation of agent i is updated to be the sum of the average $\langle\theta\rangle$ of the orientations of the other agents within a certain interaction radius $r^{\text{alignment}}$ and a uniformly distributed noise $\eta \in [-\eta_0/2, \eta_0/2]$, i.e., $\theta^{\text{alignment}}(i) = \langle\theta\rangle_{r<r^{\text{alignment}}} + \eta$. $r^{\text{alignment}}$ is set to approximately ten times the agent size l_a. The local alignment term of agent i arises as the orientation of this agent multiplied by the agent speed v_a:

$$\boldsymbol{v}_{a-a}^{\text{alignment}}(i) = v_a \left(\cos\theta^{\text{alignment}}(i), \sin\theta^{\text{alignment}}(i)\right)$$

The repulsion between the agents is modeled as

$$\boldsymbol{v}_{a-a}^{\text{repulsion}}(i) = \sum_{i \neq j} \exp\left(-\frac{\|\boldsymbol{r_{ji}}\|}{l_a}\right) \frac{\boldsymbol{r_{ji}}}{\|\boldsymbol{r_{ji}}\|}$$

with $\boldsymbol{r_{ji}} = \boldsymbol{r_i} - \boldsymbol{r_j}$ where r_k denotes the position of agent k.

The attraction to the herd center quantifies the idea that agents intent to move to the middle of the herd to avoid being captured by predators. In our model, the attraction term is independent of an agent's distance to the herd center but only depends on the agents' speed and the polar angle $\phi(i) = \tan^{-1}\left(\frac{y_{\text{cm}}-y_i}{x_{\text{cm}}-x_i}\right)$ between the agent's position (x_i, y_i) and the herd's center of mass $\boldsymbol{r}_{\text{cm}} = (x_{\text{cm}}, y_{\text{cm}})$:

$$\boldsymbol{v}_{a-a}^{\text{attraction}}(i) = v_a \left(\cos\phi(i), \sin\phi(i)\right)$$

Lastly, the repulsion of an agent from the shepherd is modelled similarly to the repulsion between to agents:

$$\boldsymbol{v}_{a-s}^{\text{repulsion}}(i) = \exp\left(\frac{-\|\boldsymbol{r_{si}}\|}{l_s}\right) \frac{\boldsymbol{r_{si}}}{\|\boldsymbol{r_{si}}\|}$$

with $\boldsymbol{r_{si}} = \boldsymbol{r_i} - \boldsymbol{r_s}$ where $\boldsymbol{r}_s$ is the position of the shepherd and $\boldsymbol{r}_i$ is the position of agent i. Based on observations of real-world shepherds, l_s was chosen as approximately 30-times l_a.

The behaviour of the shepherd is not predefined but arises from its goal to transport the entire herd to a certain target position. This goal leads to three conditions, namely (A) the shepherd should move the herd's center of mass to the target, (B) the shepherd may not loose any agents in the process, and (C) the shepherd should keep target and herd in alignment to maintain the line of sight.

These three transport requirements are weighted with $W_{\text{mean}}, W_{\text{std}}$ and W_{col} respectively and linearly combined into an objective function for the shepherd:

$$C(\boldsymbol{r_s}) = W_{\text{mean}}|\boldsymbol{\Delta r}| + W_{\text{std}}\sigma_{r_{\text{cm}}} + W_{\text{col}}|\boldsymbol{\Delta R}_{\text{col}}| \quad [2]$$

The importance of transporting the herd to the target is represented by $|\boldsymbol{\Delta r}| = |\boldsymbol{r}_{\text{target}} - r_{\text{cm}}|$, where $\boldsymbol{r}_{\text{target}}$ is the position of the target. $\sigma_{r_{\text{cm}}} = \left(\frac{\sum_i (r_i - r_{\text{cm}})^4}{N}\right)^{1/4}$ models the objective of keeping the herd cohesive and not loosing any agents. The advantage gained from keeping the flock within the line of sight of the shepherd is represented by $\boldsymbol{\Delta R}_{\text{col}} = \boldsymbol{r_s} + l_s \frac{\boldsymbol{r}_{\text{cm - target}}}{\|\boldsymbol{r}_{\text{cm - target}}\|}$ where $\boldsymbol{r}_{\text{cm - target}} = \boldsymbol{r}_{\text{target}} - \boldsymbol{r}_{\text{cm}}$.

The actual simulation is based on a forward Euler scheme implemented in C++. At each timestep, the positions of all agents are updated based on formula 1. For the shepherd, several directions are randomly sampled from the uniform distribution on $[0, 2\pi)$ and the direction corresponding to the minimal value of the objective function is chosen.

Implementation of the second shepherd. We began our study by exploring existing literature that showcased the use of two dogs for shepherding. Through this research, we found that in order to adapt the model for multiple shepherds, we needed to make two modifications. Firstly, we aimed to include a mechanism that discourages shepherds from getting too close to each other. Secondly, we wanted to adjust the way sheep and shepherds repel each other, ensuring it considered the sum of the repulsions between sheep and each individual shepherd. Although the latter was already part of the original paper's implementation, it was not functioning correctly, and we corrected the code to ensure its proper operation.

To prevent the shepherds from coming too close, we added an additional term to the cost function, which models the proximity of one shepherd to another. We define the proximity as the inverse of the distance, so two shepherds that are very close to one another are penalized more heavily. The proximity penalty for one shepherd is the sum of the proximities of all other shepherds. We introduced a parameter named `shepherd_distance_penalty`, which is used to balance the proximity penalty with the other terms in the cost function. The optimal value for this parameter is yet to be decided through experiments, but we expect it will differ between the different shepherding modes: driving, droving and mustering.

With these adjustments in place, we successfully implemented the basic version of the model with two shepherds.

Implementation of the fence. The original implementation provided by the paper's authors included a codebase featuring a fence implementation, specifically a function for calculating the repulsion force exerted by a fence on a sheep. However, due to a lack of explanation in the paper regarding the interpretation of the fence and the difficulty of understanding the author's implementation solely through code inspection, we decided not to use their code. Instead, we opted for a basic fence implementation, leaving room for potential future extensions such as incorporating an actual repulsion force from the fence.

We introduced minimum and maximum x and y coordinates, defining the boundaries of the surrounding fence. When calculating the next step for a sheep or dog, if the computed value exceeds the established fence bounds, we adjust the next step's value to the corresponding minimum or maximum x or y coordinate, preventing the sheep or dog from crossing the fence. This modification was integrated into the parameters file (`params.txt`) to accommodate fence specifications as input. Additionally, we included a condition in the `timestepping.hh` file to update the next step in the presence of a fence. As a final step, we ensure that a fence is shown in the plot by incorporating the necessary code in `trajectory_plotter.py`.

Results

Results when introducing second shepherd. The implementation of a second shepherd introduces an additional dog whose task is the guide the sheep towards the target. Figures 1 and 2 illustrate the results for trajectories with one and two shepherds. A distance penalty of 0.05 was applied. As planned, the introduction of a second shepherd is evident, and it appears to follow a trajectory similar to that of the initial shepherd but at a higher speed. Further evaluation is needed to determine whether this is always the case.

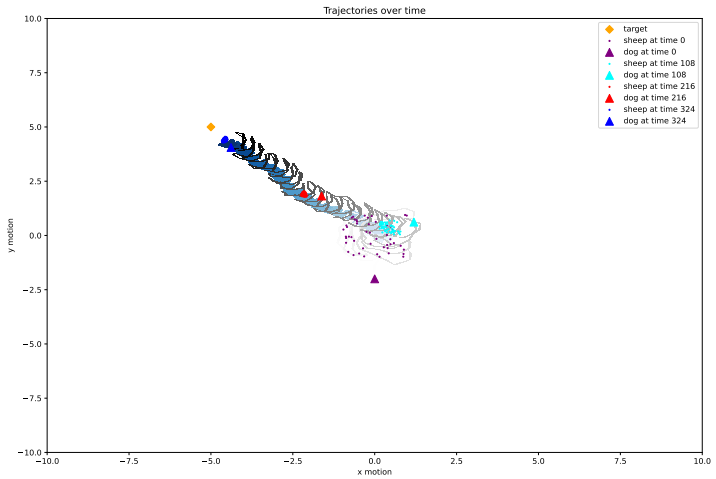

Figure 1. Trajectory plot using a single shepherd

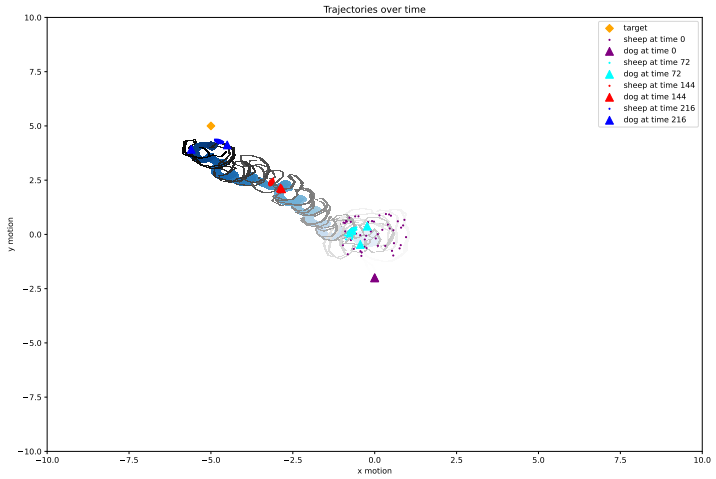

Figure 2. Trajectory plot using two shepherds

Results when introducing a fence. The implementation of a fence ensures that both the dogs and sheep are surrounded by a boundary and prevents them from crossing it. In the figures below, the outcomes are depicted for the herding style 'driving' with and without a fence. As intended, the presence of the fence effectively confines the sheep and dogs within its boundaries, while the dog still leads the sheep to the target.

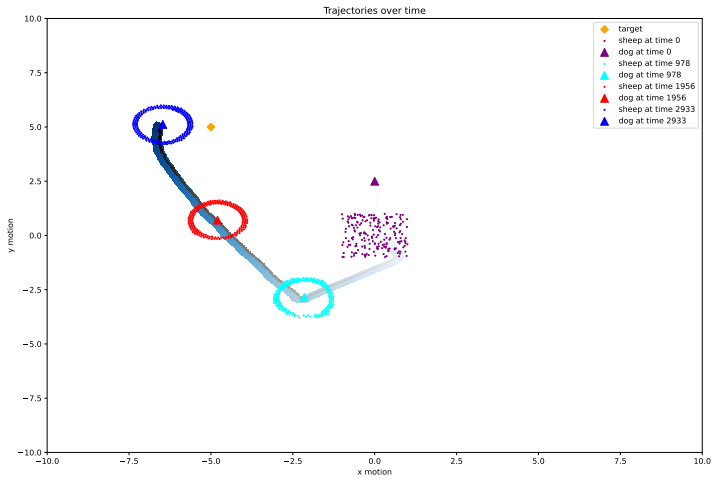

Figure 3. Trajectory plot for driving without a fence

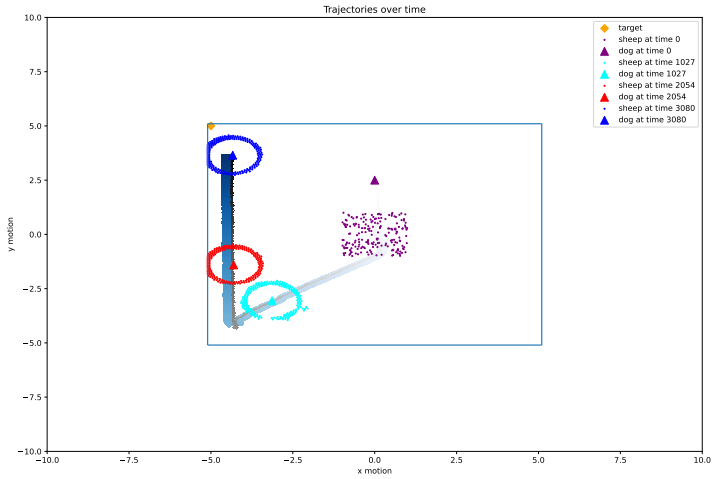

Figure 4. Trajectory plot for driving with a fence

Discussion

In this report, we've explained the model and the two additions we made. The basic versions of the extensions seem to be effective, and we've shared the results we obtained.

For the third and final report, we'll explore ways to improve our progress. This involves extensive testing with different parameters to ensure our extensions always work, making adjustments if necessary. For instance, our current testing involved only two shepherds beginning from the same location. However, we plan to explore the option of having a shepherd start from a different location and analyze the outcomes. We may also consider implementing the fence using forces, as the original authors intended. If our current implementation is satisfactory, we'll use the remaining time to implement an existing shepherding algorithm for multiple shepherds and compare the results to our model. This idea was suggested by the professor, and we still need to figure out how to approach it and which paper to use. Ultimately, we'll consolidate all our efforts from the past weeks into one final paper.

CONTRIBUTIONS. *Franziska Weber* structured the GitHub-repository, proof-read and revised the first two reports, researched existing models with multiple shepherds and wrote the model description for the second report. *Kimberley Frings* corrected and initially executed the existing implementation, implemented the model extensions with a fence and a second shepherd and composed the first two reports.*Franz Muszarsky* has not contributed anything yet.

Bibliography

1. Ranganathan A, Heyde A, Gupta A, Mahadevan L (2022) Optimal shepherding and transport of a flock.
2. Reynolds CW (1987) Flocks, herds and schools: A distributed behavioral model in *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*. pp. 25–34.
3. Vicsek T, Czirók A, Ben-Jacob E, Cohen I, Shochet O (1995) Novel type of phase transition in a system of self-driven particles. *Physical review letters* 75(6):1226.