PNAS RESEARCH ARTICLE | COLLECTIVE BEHAVIOUR

Modelling Group Dynamics and Predator Response in Meerkat Colonies

Simona Cholakova[1] and Filip Trajkoski[1]

Meerkats forage in cohesive groups, rely on rotating sentinels to watch for predators, and coordinate through distinct call types, yet few computational models capture how vigilance, warning calls and collective movement fit together. Learned models of animal behaviour, such as graph neural networks, can match the accuracy of simple baselines but have been reported to produce statistically unrealistic simulated sequences. This raises the question of whether a learned model is really better than a simple rule derived directly from what is known about the animals. To explore it, we build an agent-based simulation of a meerkat colony in Unity 2D, in which behaviour is governed by transparent rules. In this first report we describe the simulation framework we rebuilt from scratch: agents driven by a state machine, a communication system with six call types, a predator agent and a food environment.

meerkats | collective movement | quorum decision-making | agent-based model | alarm calls | simulation

Significance

Many animals, from meerkats to fish and birds, survive by moving and staying alert together, yet computer models rarely capture how groups combine vigilance, warning calls and collective decisions. We propose a simulation of meerkat colonies that links rotating sentinels, alarm calls that signal different levels of danger, and group movement triggered once enough members are ready to go. By testing whether a simple, transparent rule can reproduce realistic group behaviour better than complex learned models, this work may help us understand animal cooperation and design more interpretable models of collective behaviour.

1. Introduction

Group-living animals must stay together and stay safe at the same time. Individuals have to decide when to move, where to go and how to react to danger, usually with only local information and no central leader. Meerkats (*Suricata suricatta*) are a well-studied example. They forage in cohesive groups, rely on sentinels who take turns watching for predators while the others forage, and coordinate through a rich vocal repertoire.

Several studies describe how this coordination works. Rauber and Manser (1) showed that sentinels use discrete call types that refer to predation risk, and that this improves the efficiency of the sentinel system. Bousquet et al. (2) showed that a cohesive group of meerkats decides to set off through quorum voting. Quorum is the minimal number of votes needed to make a decision, in this case whether the group moves or not. Engesser and Manser (3) found that foraging meerkats adjust their close call rates to their proximity to other meerkats, creating vocal hotspots that attract and may guide group movement.

On the computational side, Gendron et al. (4) modelled the behaviour of social animals with causal structure discovery and graph neural networks. Their models match the accuracy of baselines, but the sequences they simulate are statistically unrealistic. This suggests that learned models alone may not capture the mechanisms behind collective behaviour. To our knowledge, no existing simulation combines sentinel rotation, referential alarm calls and quorum-based movement.

In this work we ask whether a learned model is really better than a rule derived directly from what is known about the animals. We build an agent-based simulation of a meerkat colony in Unity 2D, in which behaviour is governed by simple, transparent rules, and we compare it roughly with the learned model of Gendron et al. In this first report we describe the simulation environment we built, which serves as the foundation for the rule-based model.

2. Methods

In this section, we will describe the model that was designed based on observations made in the cited researches. The design of the model is not yet complete, and we plan to improve it using knowledge from additional observational researches about meerkats.

The base of our model is a state machine, or a deterministic finite automaton (DFA).

Author affiliations: [1]Faculty of Computer and Information Science, University of Ljubljana, 1000 Slovenia

CONTRIBUTIONS

The authors declare no competing interests.

[1]S.C. contributed equally to this work with F.T.

[2]To whom correspondence should be addressed. E-mail: sc60333@student.uni-lj.si
ft63405@student.uni-lj.si

For Meerkats we have the following states:

- Idle State
- Moving State
- Foraging State
- Run State
- Recruit State
- Mob State

The following text will explain what is done in each state, as well as the transitions between them.

Idle state - in this state is simply to model when the meerkats are stationary, possibly doing some action. One such example is when a meerkat is digging for food, which takes a bit of time to check if there is food at a specific spot.

Moving state - this state is used to model when the whole group of meerkats is moving together, looking for an area to forage, i.e. look for food. The movement of the meerkats is a modified version of Reynold's Boids algorithm. Meerkats evolved to communicate mainly through calls, due to their environment involving visual blockages and due to the fact that when meerkats dig for food, they don't have a good view on what is above ground. Hence, instead of taking the positions of the neighbours seen visually, we collect information from the close calls given by other meerkats. Separation and cohesion is modeled by whether a close call was given from either a too close or too far position. We are still struggling with the issue how to model the alignment, since in the classic boids with visual perception, each agent can see the direction in which the neighbours are heading, but this is not an information which can be easily inferred from a call.

Foraging state - is for when meerkats are not moving long distances, but rather slowly exploring an area and digging for food. Bousquet et. al. (2) observed that the transition from *foraging state* to *moving state*, happens through quorum voting. As a small reminder, quorum is the minimal number of votes needed for a group to make a decision. In this specific case, the authors observe that if at least 3 meerkats (in groups of 6 to 19) give moving call, the whole group abandons foraging the area and starts moving to another spot. The authors hypothesize a likely cause for an individual meerkat giving a moving call is possibly due to impatience, from failed attempts while digging. And that is how we model a meerkat wanting to give moving call. The movement is implemented by maintaining spacing using close calls plus some exploration in a random direction.

Note for the following 3 states. An interesting behaviour noticed in meerkats is that they don't always run away from predators. Sometimes they group around a predator, i.e. mob around him, and try to scare him away. Whether a group of meerkats will run away or mob around a predator, in our model depends on the speed of the predator. Graw et. al. (5) studied this behaviour and showed that wild meerkats will try to mob slow grounded predators (ex. snakes) and run away from fast aerial predators, as well as grounded predators who hunt through fast ambushes. Since our implementation initially focuses on 2D simulation, we design so that meerkats run away from fast predators immediately and try to mob slow predators until attacked.

Run State - is when a meerkat spots a fast predator and runs away from it.

Recruit State - when a meerkat spots a slow predator, it doesn't instantly try to mob him. Instead it first tries to recruit at least 2 other meerkats to help with the mobbing. This state is for that recruiting process.

Mob State - when enough meerkats are recruited, they surround the slow predator, stand in an upright pose and make noises to try to scare the predator away. In this process too, other meerkats are recruited as well, but in here we have a minimal amount of meerkats needed. The mobbing continues until the predator starts turning, facing in a different direction from the meerkats, slowly walking , or until the predator attacks a meerkat,

You can notice that in section 1, a few meerkats acting as sentinel guards was mentioned, and this is currently not in the design. To properly design and

implement this feature, we need to explore more researches and see if the ones acting as a sentinel are picked at random or if there are some designed meerkats to act as sentinels, or if there is a rotation system where each meerkat acts as a sentinel eventually, or there is some other observation found in nature.

Next, we will describe how we model the predator. For simplicity, we limit there being at most 1 predator on the map. There can be different predators, but only one will be interacting with the meerkats at any given time. And also, the predator has an additional variable `hunger`, which defines how willing the predator is to chase a prey, keep hunting even if mobbed, etc.

The predator is also modeled through a state machine, with the following states:

- Explore State
- Chase State
- Mobbed State
- Turn away State

Explore State - when the predator doesn't see any meerkats. It explores the map, moving at a normal pace (not too fast). It transitions to chase state when it sees at least 1 meerkat.

Chase State - when a predator chases a specific meerkat. Our model currently picks a random meerkat from the seen ones as a target to chase. Some alternative options we are considering are picking the closest one as target and chasing an average position of all the visible meerkats. The latter option doesn't seem very efficient if the group of meerkats is splitting in half and heading in opposite directions. The predator moves twice as fast compared to the explore state.

Mobbed State - this state is similar to the chase state, however the predator stands still, or moves slowly, deciding whether to keep hunting or turn away based on hunger. The more hungry a predator is, the more likely it is that it will keep chasing. If it's already satisfied (not hungry), there is a high chance that it will turn way.

Turn away State - is for when a predator decides to turn away from the meerkats who are mobbing him. The predator remains in this state until it sees no meerkats, and a bit more time moving in the same direction. Then it transitions back to explore state. This is different from the run state in meerkats, in the fact that meerkats run away from only 1 entity, meanwhile the predator is running away from multiple entities. Otherwise the same state could be re-used for both cases, from an implementation perspective

3. Results

In this section we will discuss the current status of our implementation.

First, the implementation is done in Unity in the UPR 2D pipeline with the code being written in `C#`.

Currently we haven't yet fully implemented the model as described above. The current status is the following:

- Meerkat State Machine has implemented: Idle State, Moving State, Foraging State, and Run State.
- Predator State Machine has implemented: Explore State and Chase State.

A meerkat is represented by a white circle, a predator by a red (pointy) triangle, and the foods which are collected by the meerkats are represented by a green square. In the future it's possible that we might replace these shapes with some more appropriate sprites.

The visual perception of each entity is modeled as a cone, defined by angle width and radius. We work under the assumption that a predator has tighter vision, but can see further and that a meerkat has wider vision, but can't see as far.

The call communication system in meerkats is implemented by using a Scriptable Object instance as a communication channel, called `CallManager`. So when one meerkat wants to give a call, it doesn't need to have a reference to

every other meerkat, just a reference to the `CallManager`, and can be as simple as calling the `CallManager.SendCall()` method, which will further delegate the call to every other meerkat. This can and will be improved by giving the call at appropriate times, but unless a pair of meerkats are really far from each other, a call should be heard almost instantly ($speed = 1Mach \approx 340\frac{m}{s}$).

4. Discussion

Movement. As mentioned above, meerkats can infer the position of other meerkats through the close calls they frequently give (from 4 times per minute up to 20 times per minute, depending on distance). But the direction in which they are heading cannot be inferred from a given call. One idea is that maybe each meerkat can infer this information visually, not by checking constantly, but with a some frequency, for example once every 5 seconds. Or another idea is that alignment isn't needed, just keep a proper spacing from the other meerkats, and if a meerkat is driven too far from the others, it will be pulled back via close calls. And each meerkat keeps heading in its own direction, which will be fixed by maintaining spacing.

Calls. Calls are currently delivered to every meerkat instantly, which is unrealistic. We will limit how far each call type can be heard and deliver calls after a delay that depends on distance.

Predators and sentinels. We plan to augment our implementation of predators and sentinels, so that the group's response to danger can be studied with the alarm call types described in (1). The close-call mechanism of (3), in which call rates depend on proximity to other meerkats, gives a natural basis for keeping the foraging group together.

Rule-based vs. learned models. Our main question is whether a learned model is really better than a rule derived directly from the biology. This follows from the work of Gendron et al. (4), whose models match the accuracy of baselines but produce simulated sequences that are statistically unrealistic. We therefore take a different route: instead of learning behaviour from data, we derive simple rules from the meerkat literature and test whether they produce realistic group behaviour. We plan to log the simulation and compare its behaviour with the learned model, and we expect this comparison to be a rough estimate and not a full benchmark. We will also run the simulation for different group sizes, since the groups studied in (2) ranged from 6 to 19 individuals, to see whether the quorum threshold stays constant or grows with group size, a question the original authors leave open.

Limitations. A rule-based simulation is only as realistic as the rules we put in. Choosing the parameters, such as T, β, call ranges and speeds, will require care, and we will have to validate them against reported values from the literature, not against our own expectations. The rough comparison with learned models will also say little about their accuracy, so our claims will be limited to realism of the simulated group behaviour.

Data, Materials, and Software Availability. The source code, data and all materials used in this work are available in the project GitHub repository.

ACKNOWLEDGMENTS. This work was carried out as part of a course project for Collective Behaviour at the Faculty of Computer and Information Science, University of Ljubljana. We thank Prof. Iztok Lebar Bajec for his guidance and feedback. S.C. wrote the introduction and the literature review and prepared the figures. F.T. wrote the methods section and the discussion. Both authors reviewed and copyedited the full manuscript.

1. R Rauber, MB Manser, Discrete call types referring to predation risk enhance the efficiency of the meerkat sentinel system. *Sci. Reports* **7**, 44436 (2017).
2. CAH Bousquet, DJT Sumpter, MB Manser, Moving calls: a vocal mechanism underlying quorum decisions in cohesive groups. *Proc. Royal Soc. B: Biol. Sci.* **278**, 1482–1488 (2011).
3. S Engesser, MB Manser, Collective close calling mediates group cohesion in foraging meerkats via spatially determined differences in call rates. *Animal Behav.* **185**, 73–82 (2022).
4. G Gendron, et al., Behaviour modelling of social animals via causal structure discovery and graph neural networks (2023).
5. B Graw, MB Manser, The role of mobbing in wild meerkats (*Suricata suricatta*). *Ethology* **113**, 189–196 (2007).