Chapter 1
Smart Shepherding: Towards Transparent Artificial Intelligence Enabled Human-Swarm Teams

Hussein A. Abbass and Robert A. Hunjet

Abstract The aim of this chapter is to uncover the beauty and complexity in the world of shepherding as we view it through the lens of Artificial Intelligence (AI) and Autonomous Systems (AS). In the pursuit of imitating human intelligence, AI researchers have made significant and vast contributions over decades. Yet even with such interest and activity from within industry and the academic community, general AI remains out of our reach. By comparison, this book aims for a less ambitious goal in trying to recreate the intelligence of a sheepdog. As our efforts display, even with this seemingly modest goal, there is a plethora of research opportunities where AI and AS still have a long way to go. Let us start this journey by asking the basic questions: what is shepherding and what makes shepherding an interesting problem? How does one design a smart shepherd for swarm guidance? What AI algorithms are required and how are they organised in a cognitive architecture to enable a smart shepherd? How does one design transparent AI for smart shepherding?

Keywords Explainable artificial intelligence · Interpretable artificial intelligence · Transparent artificial intelligence · Shepherding · Swarm control · Swarm guidance · Swarm ontology · Swarm tactics

The aim of this chapter is to uncover the beauty and complexity in the world of shepherding as we view it through the lens of Artificial Intelligence (AI) and Autonomous Systems (AS). In the pursuit of imitating human intelligence, AI researchers have made significant and vast contributions over decades. Yet even with such interest and activity from within industry and the academic community, general AI remains out of our reach. By comparison, this book aims for a less ambitious goal in trying to recreate the intelligence of a sheepdog. As our efforts display, even

H. A. Abbass (✉) · R. A. Hunjet
School of Engineering and Information Technology, University of New South Wales, Canberra, ACT, Australia
e-mail: h.abbass@unsw.edu.au; r.hunjet@adfa.edu.au

H. A. Abbass, R. A. Hunjet (eds.), *Shepherding UxVs for Human-Swarm Teaming*, Unmanned System Technologies,
https://doi.org/10.1007/978-3-030-60898-9_1

with this seemingly modest goal, there is a plethora of research opportunities where AI and AS still have a long way to go. Let us start this journey by asking the basic questions: what is shepherding and what makes shepherding an interesting problem? How does one design a smart shepherd for swarm guidance? What AI algorithms are required and how are they organised in a cognitive architecture to enable a smart shepherd? How does one design transparent AI for smart shepherding?

1.1 From Swarm Intelligence to Shepherding

The concept of swarm intelligence has been viewed from multiple perspectives in the literature. A biologically restricted definition [2] limits swarming to groups of social insects such as ants, termites, and many types of bees and wasps. The majority of the current literature sees swarm intelligence through a complex adaptive systems lens, and defines it as the interaction of simple agents that results in emergent self-organised behaviour, especially flocking. This latter definition usually relies on examples that include schools of fish and flocks of birds. We will assume this latter definition as a starting point in this chapter, then present a precise definition of swarming.

Consider a school of fish swimming in unison. Individual fish only have limited knowledge of their counterparts, i.e., they are unable to see the entire school, yet by adjusting their individual speed, alignment, and spacing from their nearby peers, they are able to swim cohesively.

Craig Reynolds [15] showed that a similar concept applies to the way birds flock in his seminal work on bird-like artificial objects that he named BOIDS (bird-oid object). His approach led to simulations of multi-agent systems which displayed flock-like behaviour only through local interaction with peer agents.

In classic robotics, every behaviour is fully engineered, resulting in more and more complex pieces of software for a single robot to display a reasonably complex behaviour. Swarm intelligence diverges from this concept on many fronts. First, the displayed complex behaviour on a group level can be generated from the interaction of simpler individuals. The complexity of behaviour is no longer encoded within each agent, but in the interaction space. Second, the simplicity of the rules allows an agent to respond and act fast, as opposed to executing complex software programs. Third, the computational burden of complex software programs requiring CPU and memory are eliminated, allowing agents with minimal processing power to display complex behaviours as a result of their interaction. Consequently, the energy and battery requirements are lower for these robots, allowing for extended operating time. Fourth, agents in the swarm normally rely on local information; thus, they have lesser demands on communication and complex sensors. This characteristic of the system has the ability to ameliorate congestion on the communication network.

The above discussion on swarm robotics, while encouraging, is missing two crucial requirements that are fundamental in technological solutions for most practical uses. One is related to the absence of a mission objective (how does the swarm know what it is meant to do?) and the second is related to the absence of a mechanism to instruct or command the swarm (how does one interact with the swarm?). Researchers in swarm intelligence have been innovative in addressing these gaps with approaches including external manipulation of a member within the swarm, which engenders a global effect due to the dynamic coupling amongst swarm members. Studying the impact of this form of benign or malignant control reveals that the swarm indeed could be influenced in that way [22].

Shepherding, we contend, provides a more disciplined approach to addressing both requirements. There are three basic agent types in shepherding: the human farmer, the sheepdog and the sheep. We assume a closed-world assumption, where these are the only three agent types. These agents are presented in descending order of cognitive complexity. The human farmer normally plays the role of the shepherd; the agent with the social responsibility and accountability towards the overall mission. The farmer possesses the cognitive capabilities and seeks to achieve the mission's objectives, which could range from herding the flock of sheep to patrolling them in an area away from danger.

However, the farmer does not have the physical capacity to run after the sheep, which can run as fast as 40 km/h. The fastest human speed recorded in history is close to 45 km/h[1]with the average speed in the Guinness World Record being 37.57 km/h. Clearly farmers are not expected to be top athletes. Even if they were, they do not have the endurance to sustain this speed for a long time. A sheepdog, however, can and is capable of running at speeds of up to 50 km/h.[2]The speed differential between the sheepdog, and the sheep is sufficient to offer an asymmetric physical advantage, in addition to cognitive advantage sheepdogs have over the sheep.

The above discussion creates an interesting partial order on these three agents. From cognitive ability perspective, the order is: Human $>$ Sheepdog $>$ Sheep. From a physical ability perspective, the order is: Sheepdog $>$ Sheep $>$ Human. These partial orders define the asymmetric relationship in the problem and explain the importance of a sheepdog to the human. One aspect that compounds this order is the environment, where the sheep (their location and mobility) combined with the environment (which may contain obstructions) can generate a level of complexity exceeding that, which can be addressed by the sheepdog's cognitive capabilities.

Within the shepherding task, the sheepdog acts as an actuator that augments the human physically, while the human could be perceived to augment the sheepdog cognitively. Together, this partnership is sufficient in providing cognitive and physical superiority over the sheep to complete the task successfully.

[1]https://www.guinnessworldrecords.com/products/books/superlatives/fastest.

[2]https://a-z-animals.com/.

1.2 Shepherding

The Cambridge Dictionary defines the verb to "shepherd" as "to make a group of people move to where you want them to go, especially in a kind, helpful, and careful way" [14] or "to move sheep from one place to another". [14]. We distil three characteristics of shepherding from these definitions:

1. Shepherding is concerned with the guidance of a group in some space. The concept of 'moving from one place to another' should not be limited to physical space; the mobility of an agent could occur in any space, such as land (see Chap. 10), air (see Chap. 11), sea (see Chap. 2), and information/cyber space.
2. Classical shepherding guides cognitive agents such as humans and animals, but the approach could also apply to artificial cognitive software or robotic agents. Guidance assumes that the agent to be guided is at least receptive to the guiding agent.
3. Guidance in shepherding is achieved through influencing an agent's position in a "responsible" way (see Chap. 9); that is, an agent's well-being is taken into account during the process, with guidance being "kind" and, we add, "ethical".

These three characteristics distinguish the research directions of shepherding from classic group guidance and formation control; they insinuate a level of sophistication and cognition in shepherding, requiring that actions are taken in-light of calculated risk. Moreover, it is important to have a model of the agents to be herded to modulate the influence in such a way as to be "kind" to them. In classic group guidance, the primary objective is simply to steer a group towards a stated target area. In shepherding, the welfare of the group (see Chap. 9) is explicitly stated as an objective.

As mentioned, in order to ensure influence, and therefore guidance, are carried out in a kind manner, shepherding requires the shepherd to have a cognitive model of the agents it intends to guide. This allows the shepherd to predict how the agents will respond to the action that it takes. Through feedback, it is able to adjust its action to the response of the agents it is guiding. The shepherd needs to have the cognitive capacity to infer from the state of the flock the activities and emotional (such as fear) states of the flock (see Chap. 7).

When implementing shepherding in a multi-agent framework (e.g., a robotic swarm) a level of anthropomorphism is required. Objectives mirroring those of caring and ethics must be displayed and are analogous to taking into account the dynamics of the robotic system to be herded and the associated internal states of the agents (for example, the chance of inducing collisions and the remaining energy/battery, memory and cognitive processing (CPU) levels). These objectives necessitate the design of a cognitive agent; i.e., one with a cognitive architecture which can model other agents, the environment, and itself.

In reactive agents (see Chaps. 2 and 3), the cognition of the agent is usually replaced with transfer functions that directly map information from sensors into actions for the agent to actuate on the environment, others, and self. To design a reactive sheepdog that is responsible for the welfare of the animals it shepherds, the designer needs to encode that welfare into the transfer function. For example, to avoid scaring a sheep, the sheepdog may need to stop before it violates the minimum separation it must remain from a sheep; a distance known as the stalling distance (see Chap. 4). Thus, the welfare objectives are encapsulated by reactive equations that can sense indicators about these objectives and adjust an agent's behaviour accordingly.

As discussed, in the human–animal world, farmers play the role of the shepherd. It is they who have the intent to complete the mission (shepherding tasks such as herding or patrolling) with all of its complexities and conflicting objectives (for example, move the flock as fast as possible, but minimise stress induced on the animals). In order to achieve their mission, they utilise their sheepdogs, but due to the cognitive disparity between these two agents, the shepherd transforms its intent into very precise and concrete tasks and issues commands that the sheepdog is trained to perform. If the mission is to herd the flock, shepherding offers an excellent practical application where a task is clearly decomposed into sub-tasks (e.g., collect and drive) that are used to teach the sheepdog how to shepherd.

Although they do not necessarily know of the overall mission objective, the sheepdogs have the intelligence to understand and action the commands issued by the shepherd. As cognitive agents, they can sense the sheep and determine an appropriate course of action that does not over stimulate the sheep. These task decomposition and teaching concepts offer a great deal of opportunities for robotic systems (see Chap. 10).

Despite that the concept of "shepherding" is well known and applies to human–human and human–animal interaction, it has not received significant attention from the modelling, simulation, and artificial intelligence communities proportional to its importance or potential practical impact. A recent review on the topic [11] reveals the sparse literature on shepherding, and identifies a few research directions on the topic.

Instead of repeating the review in [11], we aim to complement it by unfolding the complexity of shepherding and the opportunities such complexity offers to researchers in AI, autonomous systems, and swarm robotics. These opportunities will form a roadmap for researchers in these fields to design and develop smart artificial shepherds and sheepdogs, capable of efficiently performing cooperative shepherding tasks on a level of scale infeasible for their biological counterparts.

1.3 The Practical Significance of Shepherding

As fascinating as it is to see how nature has evolved the above dynamics complemented by intriguing mathematical and game theoretic characteristics, a discussion on the practical significance of shepherding requires us to delve deeper into the problem. The sheepdog–sheep relationship falls under the wider research area of predator–prey dynamics. Interestingly, a sheepdog is trained to look after the sheep, but the sheep react to it as a threat. What matters most to us from robotic and AI perspectives is the mathematical representation of this relationship.

A sheep is influenced by sheepdogs and other sheep. We will assume that the farmer does not influence sheep directly but uses the sheepdog for this purpose. Mathematically in this setup, a sheepdog can be abstracted as a state vector, where each element of the vector represents a parameter or attribute of the sheepdog. In its simplest form, the state vector is a position vector. Other state variables that could be added include, for example, the heading of the sheepdog relative to the sheep, body posture information, and eye contact. The state vector may include behavioural attributes such as how aggressive the sheepdog is and the sheepdog's energy level. Each sheep has a similar state vector.

This formulation allows us to model an influence on an agent as simply a change in that agent's state vector. The cause of this change in magnitude and direction of a state vector can be represented as a force vector. That is, the application of this force vector to an agent's state vector elicits a change in the agent's state. Given that multiple agents may simultaneously affect another agent, we will refer to an influence vector, which is formed by combining a number of force vectors. For example, the position information of one sheep may generate a repulsive force to avoid collision with another but also an attraction force to socialise, while simultaneously there is repulsion from the sheepdog also impacting the sheep's position.

All force vectors impacting a particular agent (a sheep or sheepdog) are summed to form the overall influence vector impacting that agent at each time step. Note that this force vector formulation allows the modelling of influence on any parameter within the state vectors of an agent, i.e., it is not limited to just position and speed, but can be used to model impact on emotional state such as fear level and fatigue. This vector representation offers significant advantages to robotic implementations in today's computing environment due to advances in Graphical Processing Units (GPUs), where orders of magnitude improvement in processing time could be achieved by cleverly structuring the problem space using tensor algebra.

The above English-based description will be transformed into mathematical equations in various chapters in this book. However, for the time being, the English description allows us to discuss the practical significance of shepherding. As a biologically inspired problem, shepherding may act as a source of inspiration to solving real-world problems; some of which are listed below:

- **Predator–Prey Dynamics:** Both real-world and simulation data could be used to understand the predator–prey dynamics between the sheepdog and sheep. Understanding these dynamics is important on both a fundamental science level and from an ecological perspective.
- **Ethics:** The original shepherding problem as defined above has two types of relationships: human–animal (shepherd–sheepdog) and animal–animal (sheepdog–sheep). When the biological sheepdog is replaced with a UxV such as a UAV (see Chaps. 9, 10, and 11), two additional relationships emerge: human–autonomy (shepherd–UxV) and autonomy–animal (UxV–sheep). Ethical

considerations are clearly required in the human–animal, human–autonomy, and autonomy–animal relationships. Shepherding therefore provides an application domain for researchers to ask and investigate ethical questions. It affords researchers the ability to run simulations and ethically approved, low-risk experiments in both synthetic and real environments [25].

- **Swarm Robotic Guidance and Control:** This is the most common use of shepherding in this book, where the sheepdog–sheep relationship offers a scalable model for swarm guidance. The sheep have been classically modelled as BOIDS, with one extra force representing a repulsive force from the sheepdog. This force vector could be modulated based on the sheepdog's proximity to the sheep [18]. The mere fact that the sheep respond to a sheepdog establishes dynamic coupling between the two agents, where one agent (the sheep) responds to the actions of the other agent (the sheepdog), putting the latter in a position to control the former. Moreover, shepherding offers an approach where a single sheepdog can manage a large number of sheep. The more disciplined the sheep are, the more of them a single sheepdog can manage. This makes shepherding a perfect model of inspiration to scale up swarm control, allowing few agents to control a larger number of agents.
- **Cyber Security:** The previous point made the implicit assumption that robots are physical entities such as unmanned aerial, ground, surface, underwater, or space vehicles. However, the mathematical models for shepherding are application agnostic and can be applied in abstract spaces modelling the information and cyber domains. The concept of mobility is conceptual and it can represent mobility of pieces of information in an abstract space, ideas in a community, people in a social network, or pieces of software in a computer network. Here the concept of shepherding becomes how one uses a few number of decision points to herd information through a network, ideas through a social network, or mobile intrusion detection systems (such as immune inspired intrusion detection systems) through a computer network.
- **Mission Cryptology:** In shepherding, the sheep do not know the intent of the sheepdog, neither do they know the intent of the farmer. As such, the goal is replaced with a timeseries of influence vectors that in their totality achieve the overall mission intent. Any of these influence vectors in isolation is insufficient to decode the intent of the mission. This form of encoding could be used to secure the plan and objectives of the mission, as agents do not need to explicitly communicate anything but a series of mathematical vectors. Moreover, the vector representation enables a further level of security via encryption, although in highly dynamic environments, this should be balanced against the associated increased processing time.
- **Human-Autonomy Teaming:** The human shepherd interacts with a manageable number of entities, which in turn exercise control over a large flock. The cognitive load on a human operator is clearly dramatically reduced from that required when explicitly controlling the individual members of the flock. The social interaction, cognitive load, trust, and the wider human factors and cognitive engineering considerations are worthy of further inquiry. The shepherding problem offers a

great deal of opportunities to explore some of the very fundamental problems in human–autonomy teaming, human–swarm interaction, and human–robot interaction (see Chaps. 12 and 13).

Each of the potential practical uses of shepherding discussed above raises the question of how to implement appropriate cognition within an artificial agent, such as a robot, to achieve the behaviours required in each problem. In the next sections, we will discuss reactive and cognitive designs of a shepherding agent.

1.4 Reactive vs Cognitive Shepherds and Sheepdogs

Different schools of thought exist on how to model an agent (see [24] for a detailed introduction on agents). Those prescribed to the reactive school transform humans' understanding of how the system should work into equations, rules, or models that directly associate a group of responses to a group of stimuli. One particular branch of research in reactive agents is Physicomimetics [20], whereby the world, be it social, biological, or cognitive, is modelled using physical principles and equations.

A model within a reactive agent could be seen as the set of shortcuts, sometimes represented as a set of event-condition-action statements, used to approximate stimuli-response or cause-effect relationships. The effort sits on the shoulders of the human designer to create the model and/or parameterise it. The human system designer is required to have sufficient knowledge of all conditions and inputs that could affect the agents' states and assign appropriate actions to these Event/Condition/Actions (ECA) tuples. These ECA tuples must appropriately capture all possible behaviours required by the agents. As tasks become more complicated, such approaches clearly do not scale well; further, interactions between the agents may lead to the emergence of unintended and detrimental system-level behaviour. Recent research in reactive behaviour uses AI to estimate the structure and/or parameters of the reactive model [17]. Such an approach might be considered similar to behavioural psychology with external observed stimuli and agent responses giving insight to agent behaviour.

Conversely there also exists the cognitive school of agent modelling, similar to cognitive psychology, whereby the agent is understood in terms of its cognitive components such as executive control functions and long term memory. The cognitive school attempts to equip agents with the processes to acquire knowledge, learn, plan, and adapt; as such cognitive approaches seek to instil a greater level of autonomy in the agents.

Cognitive agents tend to rely on more complex models, and therefore more computational resources, than their reactive counterparts. They are theoretically more robust to change in the environment and/or mission than reactive agents. Reactive agents are usually much faster and are more suitable for platforms with low computational resources. The implementation of a robust agent capable of operating in complex real-world settings would generally be a hybrid, with an appropriately chosen point of balance sitting between the two extremes of totally reactive and completely cognitive. A common design approach for such a control system utilises the required time-scales for responses (i.e., the decisions to be made) to determine the switching point between cognitive and reactive control. For example, in controlling the mobility of a robotic system, actions such as immediate collision avoidance could best be achieved using an ECA reactive approach to allow the agent to respond in a timely manner to an immediate collision. In contrast, determining an appropriate path to traverse to avoid known obstacles within an area would generally use a cognitive planning approach. Additional parameters, such as computational resources available to the agent, and complexity of the missions the agent is designed for, should also be considered when determining where this balance point between cognitive and reactive implementation lies.

In terms of academic critique within the literature, the reactive school is normally criticised due to the simplicity of its models, with most models sitting at a very high level of abstraction compared to classic control theoretical or cognitive AI models. This criticism is sometimes due to a fundamental misunderstanding driven by looking at reactive models as the complete solution to a problem. This criticism was discussed by Reynolds [16], the author of the infamous, reactive *Boids* concept, who proposed three modelling levels. The highest level is concerned with action selection, requiring strategies, goal setting, and mission-level planning. This highest level sends goals to the second level, which is concerned with path selection and planning. This second level sends the calculated path to the lowest level for locomotion to actuate. Here, we see that an early pioneer of reactive rules for swarm control understood where such rules should be utilised within a hybrid system. Such approaches essentially seek to ensure that the right reactive behaviour is selected at the appropriate time, a concept adhered to in many works [7, 19].

1.5 Swarm Ontology for Transparent Artificial Shepherding

While emergent behaviour of a large number of biological entities may be modelled and achieved through simple reactive local rules, these equations are clearly insufficient to capture the cognitive capabilities of biological agents. Consider, for example, a bird; although flocking can be represented and modelled by the equations which govern BOIDS, these equations are unable to sense and calculate the parameters required by the equations or map their outputs to actions through the bird's body.

The bird uses part of its brain to sense and perceive the environment, comprehend and project each instance it is situated in, decide what to do, and then execute this decision by actuating (flying). The brains of biological agents have evolved to cluster mental processes close to each other to reduce the energy required to perform these processes, forming a cognitive architecture of sorts. We contend that to appropriately embody swarming into agents, be they robotic or software, such a construct is required. This cognitive architecture organises the mental processing within artificial agents. In order for the swarm to be able to team with a human operator, this cognitive architecture, whether it be simple or complex, should be transparent, enabling the human to learn from it, use it, and to assure its trustworthiness.

While a discussion on transparency goes beyond the scope of this chapter, designing a swarm ontology for shepherding is an important enabler for transparency. An ontology has a few key advantages in this context:

- It can provide explanation on how an agent makes decisions using concepts that a human is familiar with, especially in a particular operational context.
- It allows mapping of an explanation from one form to another; thus, the same explanation could be interpreted differently based on the user's area of expertise. The ontology connects concepts between users; thus enabling interpretable decision making, whereby the decision is transformed from the representational language of one agent to a representation that can be understood by another agent.
- It can guide a machine learner (see Chap. 6) to constrain its search space, thereby gaining efficiency in learning time, and/or ensure that what an agent learns is interpretable by another agent.
- It can be used by a human to diagnose undesirable behaviours that an artificial agent may have expressed, and/or assure its performance and suitability in an operating environment.

We present our proposed shepherding ontology in Fig. 1.1. Such an ontology needs to capture a number of elements across the different layers the system is operating on.

First, it needs to capture the behavioural set of a single sheepdog and the skills that need to be acquired by a sheepdog to perform certain actions (see Table 1.1). For example, to drive a cluster of sheep to a home point, some of the basic skills required by a sheepdog include the ability to:

- locate itself relative to the goal
- approach the sheep
- collect indicators on its performance to identify appropriate corrective actions
- . . . and so forth

This behavioural set indicates to the shepherd the capabilities of an individual sheepdog. In the absence of knowing these concepts and their associated effects, a human is unable to know what to expect from the sheepdog when it responds to a command from the shepherd. Defining the capabilities of an agent in such a way also allows an observer to label and categorise the actions that the agent performs, and, as such, enables activity recognition (see Chap. 7).

Second, the ontology needs to capture the tactics each individual is capable of (see Table 1.2). A tactic is a series of organised behaviours displayed by an individual to achieve an intent. For example, singling out a specific sheep (removing a particular sheep from the flock) may require the sheepdog to first seek that sheep as it is standing, pursue it as it starts to run, then offset pursue it as it approaches the target area so that the sheepdog does not enter that area.

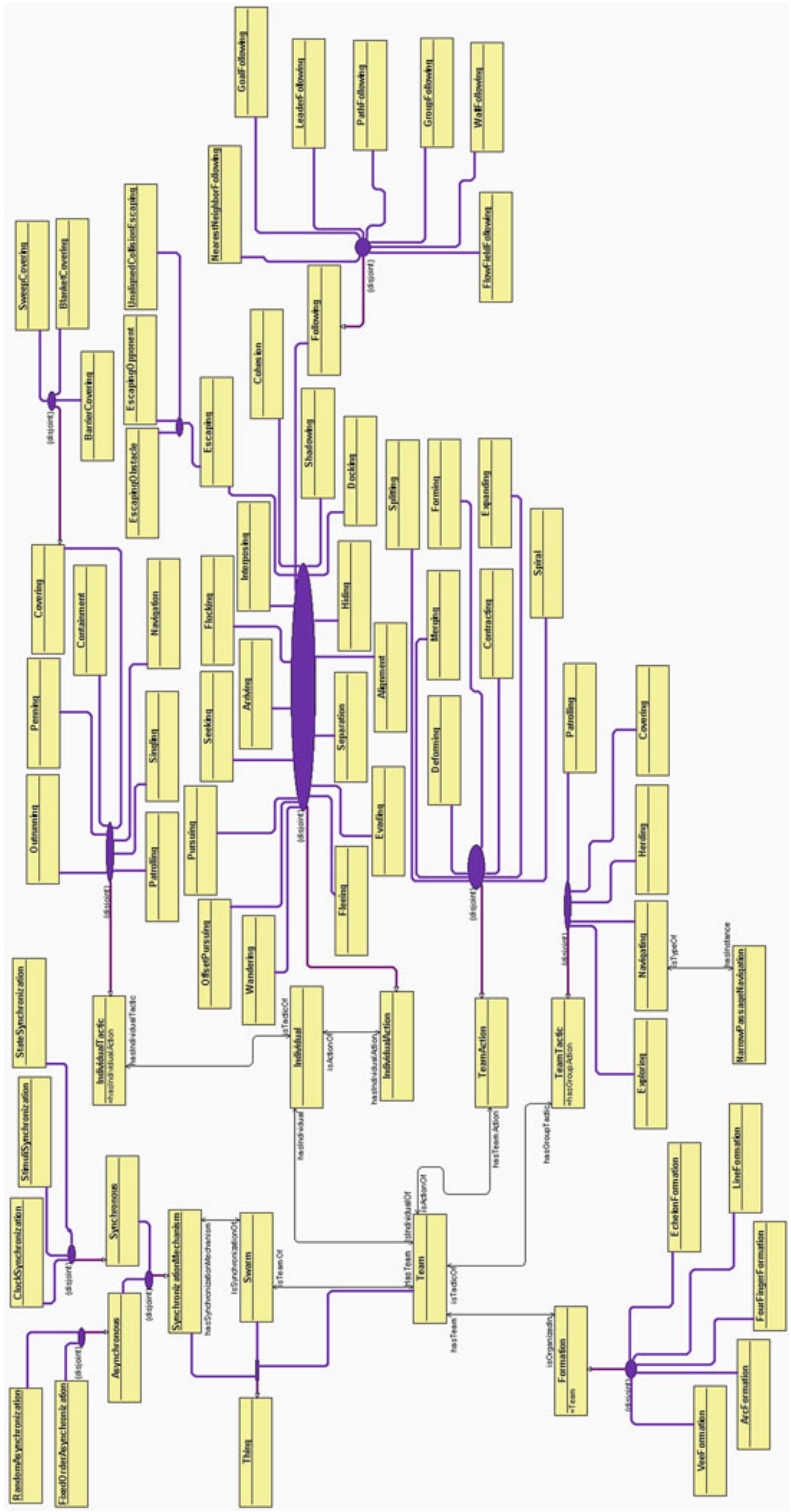

Fig. 1.1 A swarm ontology for shepherding

Table 1.1 Individual actions

Action	Definition
Wandering	A random walk.
Docking	Constrained orientation.
Evading	Steering away from a moving target.
Fleeing	Steering away from a static target.
Hiding	Set the target location to be behind an obstacle on the opposite side of an intruder and seek this location.
Seeking	Steering towards a static target, producing motion similar to that of a moth buzzing around a light bulb.
Pursuing	Steering towards a moving target. Similar to seeking with the static target position replaced with the predicted target position.
Offset pursuing	Steering near but directly into a moving target. Basically, the predator pursues the prey while maintaining a distance from the prey.
Arriving	Seeking with decaying speed as the agent approaches the target. The speed is 0 at the target location.
Interposing	Predict the centre of gravity among future positions of two or more agents then seek this position.
Escaping obstacle	Similar to flee but only when the obstacle is on a collision path with the agent.
Escaping opponent	Similar to flee but only when the opponent is on a collision path with the agent.
Shadowing	Approach the agent then align to match speed and heading.
Cohesion	Seeking the centre of gravity.
Separation	Steering away from nearby agents.
Alignment	Steering vector is the difference between the average velocity of neighbours and the agent's velocity.
Flocking	A combination of separation, cohesion and alignment.
Nearest neighbour following	Approaching the nearest neighbour
Goal following	Approaching the goal
Leader–follower	Arrival towards the leader. If in front of the leader, steer away sufficiently then resume arrival. Maintain separation to prevent crowding.
Wall following	Approach a wall then maintain certain distance.
Path following	Once a path is defined, margin is added to create a corridor. Path following is flexible alignment to the centreline of this corridor, which is a containment within a cylinder around the path's spine.
Flow field following	Alignment with a vector field. The flow field is a cloud of points in space. Each of these points is the tail of a unit vector, representing the direction an agent should follow when it reaches that point.
Group following	Approaching the majority of the group.
Unaligned collision escaping	Predict future locations of neighbours and accelerate or decelerate to get to the expected collision site before or after the intruder (thereby avoiding the collision).

Table 1.2 Individual tactics

Tactic	Definition
Outrunning	Moving in a pear-shaped trajectory with a wider arc as the sheepdog approaches the sheep until it reaches the point of balance; the latter is the point where the stressors on the sheep due to the sheepdog position places the sheep in a state of alert, while being at the edge of moving.
Penning	The flock is driven to a small enclosure. The door is closed when all sheep get inside the enclosure area.
Singling	A specific sheep is separated from the flock.
Patrolling	Preventing a group from exiting an area or keeping it at a fixed distance.
Covering	Navigation to multiple goals.
Containment	Motion of the agent is constrained within a region.
Navigation	Localisation in the environment and identification of appropriate routes.

With actions of a single sheepdog defined, let us turn our attention to what they may do as a team.

Definition 1.1 A team is a group of organised individuals joined together to execute team-level tactics and actions.

The definition above lists four concepts related to a team: organisation (such as a formation), team tactics, team actions, and the individuals making up the team.

Definition 1.2 A formation is a spatial organisation of a team of individuals.

Definition 1.3 A team action is a basic building block of what a team can do and is capable of generating an effect/outcome.

Definition 1.4 A team tactic is an organised set of team actions to achieve an intent or a higher-order effect.

Formations come in many forms. In this chapter, we limit these to some basic formations that could apply in land, air, and sea. The ontology in Fig. 1.1 lists five basic formations: vee (individuals are organised similar to the letter "V"), arc, line, echelon, and four-finger formations.

Team actions and tactics require the involvement of at least two individuals. Tables 1.3 and 1.4 define the team actions and tactics used in the ontology, respectively. Many of these definitions were synthesised from a variety of literature. Sometimes the concept is used in a manuscript without formal definition; we address this here by introducing such definitions. Occasionally different manuscripts use the same concept but refer to it by a different name; here we look to standardise the terminology. The team actions mostly derive from [12] and the individual actions mostly from [16]. The tactics are mostly from the rules of shepherding competitions and other sources including [4, 9]. We attempt to standardise the concepts, define them, and disambiguate their use to ensure consistency across the ontology.

Table 1.3 Team actions

Action	Definition
Splitting	The team needs to split into two or more clusters.
Merging	Two or more of the team clusters combine such that the radius of the combined team is less than a threshold.
Forming	The team moves to a specific formation.
Deforming	The team breaks formation into random organisation.
Contracting	The team reduces its radius.
Expanding	The team expands its radius.
Spiral	The team moves in three dimensions around an object as a spiral.

Table 1.4 Team tactics

Tactic	Definition
Exploring	Cover a region.
Foraging	Wander with resource seeking. Classically, the resource of interest in foraging is food.
Herding	Collecting and driving a group of agents from one location to a target area.
Covering	Team navigation to multiple goals.
Blanket covering	Statically arrange the team to maximise the detection rate of the opponents (predators or prey) in the covering area.
Barrier covering	Statically arrange the team into a barrier to minimise the probability of undetected opponent (predator or prey) penetration through the barrier.
Sweep covering	Move the team across an area to maximise the number of detections in the coverage area. For example, the shepherd sends the sheepdogs to the bush in a coordinated fashion to maximise the number of sheep detected in the coverage area.
Patrolling	Team prevention of another team from exiting an area or keeping it at a fixed distance.
Navigating	Team localisation in the environment and identification of appropriate routes.

Note that, the existence of a team does not guarantee coordinated behaviour. For example, in order for birds to flock, a team needs to swarm through appropriate synchronisation to create spatial and temporal alignment of the actions and tactics of the individuals.

Definition 1.5 A swarm is a team with actions of the individuals that are aligned spatially and/or temporally using a synchronisation strategy.

The above definition gives the entry point to defining a swarm with two main concepts: the concept of a team and the concept of synchronisation strategies, where what we refer to here is behavioural synchronisation, or the ability of the team members to produce actions at the same time.

Behavioural synchronisation normally falls into two categories: asynchronous and synchronous. The former requires individuals in the team to perform an action sequentially, or independently of what another agent is doing. Examples of asynchronous strategies include using a random order to perform actions. Another example may be taking input from one agent which triggers an action in another. Consider the case of a group of humans organising themselves in a line. Each individual may rely on the person standing to their right; when this person is stable, the next aligns with that person. This right-to-left ordering is based on the state of the person on the right and independent of any clock. It will trigger an action-chain from right to left. This is an asynchronous behaviour, in that the action of the individuals does not occur at the same time. We delineate here between behavioural and algorithmic synchronicity, where the latter seeks to ensure the order of operations at the algorithmic level; i.e., note that the asynchronous behaviour of humans forming a line is achieved here by using a synchronous algorithm where action in an individual is triggered by a state change of the person on the right.

In the synchronous approach, all team members perform the action simultaneously. This form of synchronisation requires team members to have a common point of reference to know when to perform the action. One popular approach is to synchronise the clock in each agent on a common clock and the actions are performed when the clock reaches a specific pre-agreed time. Another approach is to trigger the actions across multiple entities based on stimuli in the environment such as an event (e.g., farmer's whistling) that triggers all agents to act simultaneously. These are basic approaches to get a team to synchronise actions, but more sophisticated ones could be developed from these basic principles.

We do not seek to design shepherding based swarming systems for the sake of swarming, rather the swarming behaviours should be pertinent to achieving mission success. As such, the members of the swarm need to have mechanisms to represent and execute missions meaningfully with the behaviours elicited from the interaction of the individuals appropriate for the current environmental state. Smart swarming systems need to decide how to modulate their behaviours and control when to exhibit a particular swarming behaviour. To organise these mechanisms to execute a mission meaningfully, we need an architecture for shepherding to capture the different cognitive processing required by a sheepdog to receive and process information, make decisions, and actuate on the environment. We present such an AI architecture in the next section.

1.6 Artificial Intelligence Architecture for Shepherds and Sheepdogs

Cognition may be considered a system-of-systems. An artificial agent needs to be equipped with many skills to perform the simplest functions that a biological system performs. For example, for a sheepdog to detect that there is a sheep away from the flock, the sheepdog needs first to have sensors such as eyes, ears, and nose to see, hear, and smell where a sheep is, respectively. Each of these sensors offers the sheepdog different information, such as relative position of the sheep to the sheepdog, the body posture of the sheep and orientation of head, the state of fear as displayed by changes in body odour, and more. These cues are received by the sheepdog's brain in the form of signals that need to be transformed and encoded into actionable information.

The sheepdog takes the actionable information and matches it to its accumulated experience in its long term memory and natural instinct. The sheepdog then calls upon its *knowledge of self*, comprising its available actions, to select an appropriate course of action which it may actuate on the environment to achieve its intent.

What sounds to be natural instinct seemingly called upon and executed momentarily, when appropriately analysed, requires a series of skills that are either encoded in the sheepdog's innate knowledge due to evolutionary learning and natural selection, or acquired skills from the personal experience of the sheepdog in a particular habitat. Replicating such performance in an artificial agent is a tricky proposition. The designer is required to make a series of AI algorithms available within the artificial agent to enable the acquisition, adaptation, and adoption of the required skills. These could be algorithms to detect, classify, recognise, identify, verify, and track an object; algorithms to perceive, comprehend, and project events and situations; algorithms to localise the agent, map the environment, detect and avoid obstacles, plan the path and trajectory to navigate from one location to another; to control and decision making algorithms to move, direct sensors, communicate, negotiate, influence, shape, and actuate on self, other agents and the environment.

A designer might start with some of the modules, such as perception and decision making, and implement them with simple algorithms, then gradually add more sophisticated approaches to ensure a fully functional AI that is capable to adapt and handle unseen situations. One needs only to examine the growth in the literature of articles pertaining to AI to reach the conclusion that advances and contributions to the field are occurring at an astonishing rate. To appropriately endow the agent with the ability to cope with such complexity, the cognitive architecture of an artificial agent may need to host a variety of AI algorithms for different cognitive functions and different contexts.

We propose an architecture for shepherding and sheepdogs, and present it in two stages. The first stage will focus on the decision making architecture responsible for coming to and enacting a course of action. We then expand the architecture by incorporating perception, situation awareness, and situation assessment.

The architecture is applicable to both shepherds and sheepdogs; although we have stated that the work within this book seeks to create an artificial sheepdog, we present here also the concept of expanding this work to enable the command of the artificial sheepdog by an AI (albeit, scoped down to the purpose of conducting a shepherding mission) with the primary difference between these agent types being individual cognitive capacities, including memory and processing power, and consequently, the level of sophistication of the AI algorithms to be implemented. Although both are cognitive agents, the sheepdog has limited capability to comprehend complex situations when compared to a shepherd. The practical implication of this design choice is that the sheepdog will likely rely more on reactive models to allow it to respond fast, while being light weight. The artificial shepherd needs to be a cognitive agent with the capacity to represent and monitor the evolution of the context, track progress towards achieving the goal, plan ahead, and command the sheepdog to take corrective actions to steer the trajectory of events towards one which the shepherd believes will lead to a successful mission.

We do not underestimate the amount of time it will take to automate the cognitive functions required for real-time autonomous shepherding in all contexts, terrains, and biological agents (sheep, cattle, horses, or even rabbits). However, the cognitive architectures presented in the following subsections will enable the housing of the sophisticated algorithms we will continue to iteratively develop to achieve the technological goal.

1.6.1 Shepherds and Sheepdogs Autonomy Architecture

We expand Reynolds three-level model into a five-level AI architecture suitable for modelling both shepherds and sheepdogs in Fig. 1.2.

The top level of the architecture is concerned with the mission description language. Here, a mission is defined in terms of its objectives, the available resources and any mission-specific constraints such as the required mission completion time. This contextual information is sent to the goal planner to decompose the overall mission objective into sub-goals. The sub-goals are sent to the behaviour selection level to identify and extract from the behaviour database

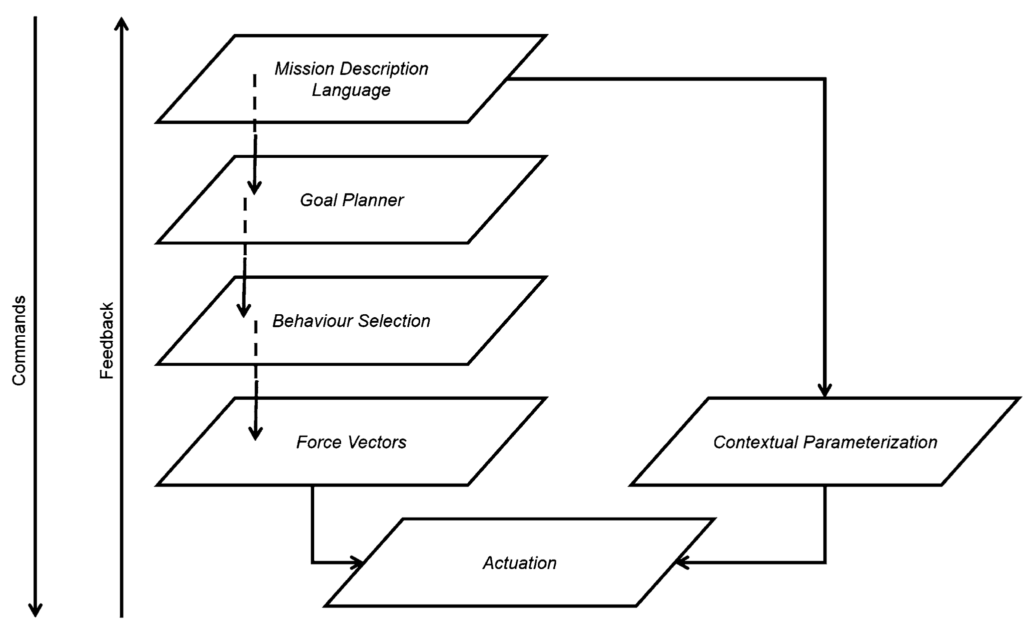

Fig. 1.2 An artificial intelligence architecture for the decision making functions for shepherding

the subset of behaviours required to achieve the set of sub-goals. The behaviours are transformed into force vectors and parameterised through the contextual parameterisation module. In this step, the vectors are combined to produce the aggregate trajectory of behaviours required to achieve the intended sub-goal. The information is sent to the low-level controllers to actuate on the environment.

We will illustrate the architecture using a smart artificial shepherd example. The mission could be stated as "Collect all sheep and bring them to the goal location within 30 minutes from now, while not stressing the sheep, using two sheepdogs of type A and three sheepdogs of type B". In the first part of this mission statement, we have the classic shepherding problem, whereby the objective is to collect and drive the sheep to a goal location. The second part sets a time constraint and an animal welfare objective. The third part defines the available resources for the mission, which in this particular case, we have five sheepdogs of two different types (note this may also be considered as a resource constraint).

The mission statement has all the information needed by the shepherd's goal planner to decompose the mission objective(s) into sub-goals that could feasibly be achieved given the constraints on the mission and available resources. The goal planner may output a sequence of sub-goals such as collect all sheep at location X within 20 min, drive the sheep to the goal within 7 min, then patrol the sheep to ensure they do not leave the goal for 3 min. These last 3 min could be allocated for contingencies if the first two sub-goals are delayed due to the uncertainty in the response of the sheep.

These three sub-goals are sent to the behaviour selection module. To collect the sheep, the module may select four behaviours: sense outlier sheep,sheepdog formation selection, sheepdog positioning behaviours, and specific pursuing behaviours for the sheepdogs to use to collect the sheep.

These behaviours are represented with attraction–repulsion equations that need to be parameterised and fused together to identify the aggregate force vector that a sheepdog needs to follow at each moment of time. To enable this, the individual force vectors associated with the behaviours are passed from the force vector module to the actuation module. Next, the contextual parametrisation module decides on the parameters and fusion weights based on contextual information such as mission resource constraints, the performance envelope of the sheepdogs and sheep in the environment, and the remaining energy for each sheepdog. This parametrisation information is also passed to the actuation module, which is responsible for using this information to appropriately weight and combine the individual force vectors to form the aggregate force vector. At this point the actuation module applies any actuation constraints and drives individual actuators commensurate with this aggregate force vector.

Figure 1.3 maps out the example used throughout this section on key components of the autonomy architecture.

The above example showcases an architecture that is purely focused on the fewest number of components required for a smart shepherd and sheepdog to make decisions. However, agents make decisions based on their perception of the environment. To transform this decision making focused architecture to operate on a

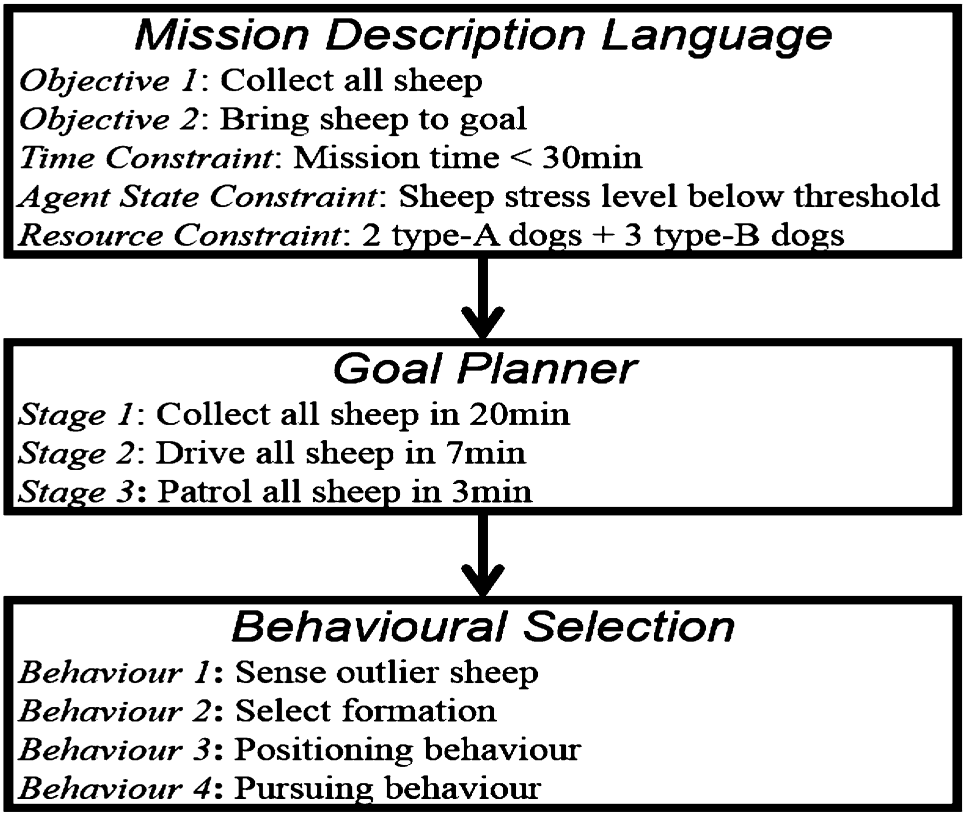

Fig. 1.3 A shepherding example using key components of the autonomy architecture

real robot, the shepherd and sheepdogs need to have the ability to sense and perceive the environment, including sensing one another. In addition, they need to be able to assess the risks associated with different parameterisations of the behaviours generated by the decision making architecture. Such assessment should consider the current situation, the overall mission context, and the sub-goals to be executed. The next sub-section will focus the discussion on the sensing and situation awareness architecture.

1.6.2 Shepherds and Sheepdogs Contextual Awareness Architecture

The contextual awareness architecture is presented in Fig. 1.4. It consists of three primary modules: situation awareness responsible for perception, comprehension, and projection; situation assessment focusing on activity recognition, course of action selection, and impact analysis; and context-driven parameter settings to parameterise the force vectors in the autonomy architecture based on the context, including the current situation.

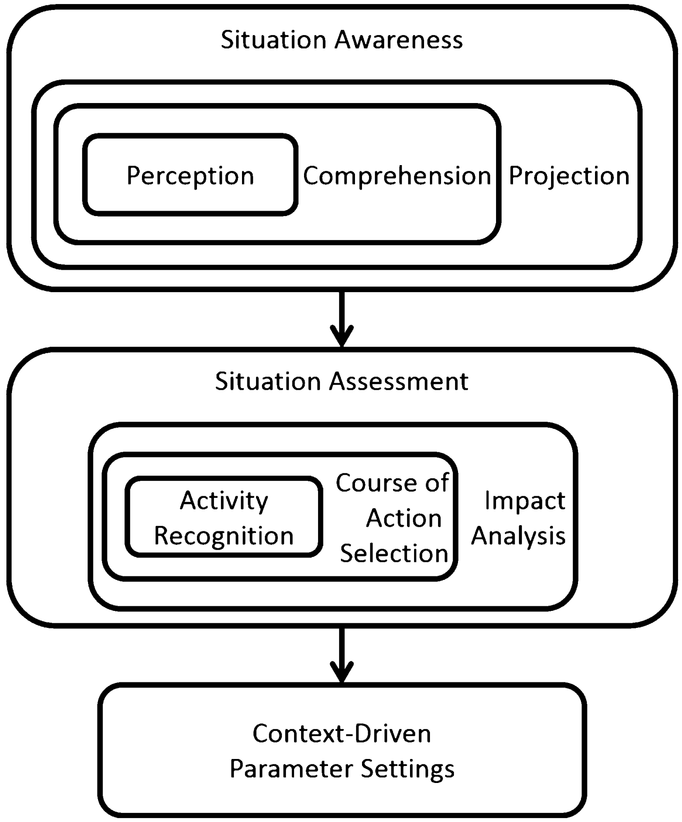

Fig. 1.4 A contextual awareness architecture spanning sensing, perception and context-driven parameter setting

Perception, on a functional level, transforms sensorial data streamed from the agent's sensors to features and indicators that the agent could act on. Comprehension transforms one or more features and indicators into summary statistics and flock- and context-level state information. Projection uses this information to anticipate and predict the evolution of these states into the future.

For example, a shepherd senses the location of sheep. The perception module transforms these raw data into features, such as the global centre of mass of the herd, the local centre of mass of the largest cluster in the herd, the relative direction between each of these two centre of masses and the goal, the location of the furthest sheep from the herd outside the largest cluster, and the relative direction between that sheep and the global centre of mass.

The comprehension module takes these features as inputs to calculate different flock-based state information such as "the herd is clustered" and "there is a stray sheep". The projection component is able to use this state information to estimate, for example, that the stray sheep will reach inaccessible bush area if it continues its current path for three minutes, or that the clustered herd will fragment into a large number of smaller clusters if left as is for ten minutes. Projection works on each state independently. The situation assessment module works on the interaction of these states.

The situation assessment module consists of three components: activity recognition, course of action selection, and impact analysis. Activity recognition is responsible for taking the features and state information from the perception and comprehension sub-modules and processing them to recognise the behaviour and/or functions and tasks an agent is undertaking. In essence, activity recognition transforms data of the individuals, and statistics of the group, to an understanding of the intent of both the individual and the coordinated behaviours of the group, as well as higher-order state information at a flock-level.

An activity recognition system applied to sheep data alone may indicate that the sheep are foraging, eating, or stressed. Other indicators could be produced by the activity recognition system such as confidence levels on each recognised activity or estimates of the parameters associated with an activity. For example, "the sheep are eating with a *high* stress level due to the sheepdogs standing in the vicinity of the sheep"; or "with a *low* stress level due to absence of a visible threat; the sheepdogs are standing far away and outside the field of view of the sheep". These examples highlight the importance of explainability in the activity recognition system, to indicate the reason an activity has been recognised and the confidence level associated with an activity.

The activity recognition system applied to the sheepdogs would produce information on the sheepdog level. For example, it could indicate that "sheepdogs are positioning themselves into an arc formation", or "the sheepdogs are running back to the water due to being thirsty". When the activity recognition system is applied to both the sheep and sheepdogs, the system attempts to recognise the reciprocal dynamics between the two agent types. For example, it could indicate that "the sheepdogs are driving the flock", or "one sheepdog is on its way to collect a stray sheep".

The information from the activity recognition system is key to inform the *course of action selection* module. A course of action in this module primarily represents the selection of strategies required to achieve an agent's intent. It is not an operational or tactical action. Note that, these courses of action are decided within the autonomy architecture. The contextual awareness architecture needs to operate ("think") on a higher level of abstraction sufficient to represent the strategic thinking of the agent, supporting it to evaluate the strategies required to achieve the mission objectives and goals. Increasing fidelity in the contextual awareness module should be done with utmost care due to the associated increase in computational resources and resulting reduction in response time.

The course of action selection may include course of action generation abilities. However, for simplicity, we will assume here that all courses of action are pre-designed and available for the agent to select from.

Continuing with the examples of the outputs of the activity recognition system discussed above, the course of action selection module may have a portfolio of high level strategies to select from such as: "fetching" or "singling".

The impact analysis module uses the current situation awareness picture formed by the agent and the activities that the agent has recognised the other agents are undertaking to assess the impact of the selected course of action on the mission objectives and goals. The impact analysis module may be tightly coupled with the course of action selection module and be part of the rationale for selecting a particular course of action. A computationally less intensive implementation would keep the two modules decoupled; in this simplified form, course of action selection may be driven by a simple decision tree, with the impact analysis module evaluating the resultant course of action without necessarily changing it, as required changes could be made at the next iteration through the contextual awareness loop.

In the case that the selected course of action is, for example, "singling", examples of the outputs from the impact assessment module could be the singled sheep is a leader amongst the flock and removing it will increase the complexity of the driving task, or that the singled sheep is an unsettled leader that will decrease the complexity of the driving task. In essence, the impact assessment module offers the agent with the ability to assess consequences and risks.

Figure 1.5 maps out the example used throughout this section on key components of the contextual awareness architecture.

The two architectures discussed so far do not touch on the communication and commands exchanged between the shepherd and sheepdog. In essence, a command that is issued from the shepherd to a sheepdog is an order to achieve a sub-goal that the shepherd decided should be executed by the sheepdog. The shepherd and the sheepdog, due to the difference in their cognitive capacity, operate at differing levels of mission complexity and over different time-scales. The command issued by the shepherd to the sheepdog (which is a sub-goal of the shepherd's mission) appears as the sheepdog's mission in its instantiation of the architecture. These commands do not require any changes in the architecture. The issuing of a command is a sensorial input to the receiving agent and action actuated by the commanding agent. In the next section, we bring these two architectures together to form the cognitive architecture for both the shepherd and sheepdogs.

1.6.3 Smart Shepherds and Sheepdogs Overall Architecture

To implement a smart shepherd and/or sheepdog, the designer should take a system-of-systems approach to structure the design of cognition in a nested and modular fashion. The architecture presented in this chapter uses modularisation to ensure a neat, efficient, and easy-to-maintain design. Nevertheless, there will always be bridges between modules in the system to pass essential information. These bridges should be minimised to reduce system complexity, but cannot be eliminated without negative consequences on the performance of the system. They provide the feedback loops which enable modules to make informed decisions.

Figure 1.6 highlights the links between the autonomy and contextual awareness architectures, and introduces the sensors and actuators. Seven links in total have been added in the figure, two of them are links to sensors and actuators. The interdependency between a module in the autonomy architecture and a module in the contextual awareness architecture is necessary for the agent to function as a smart agent, while maintaining a well-designed modular cognitive architecture.

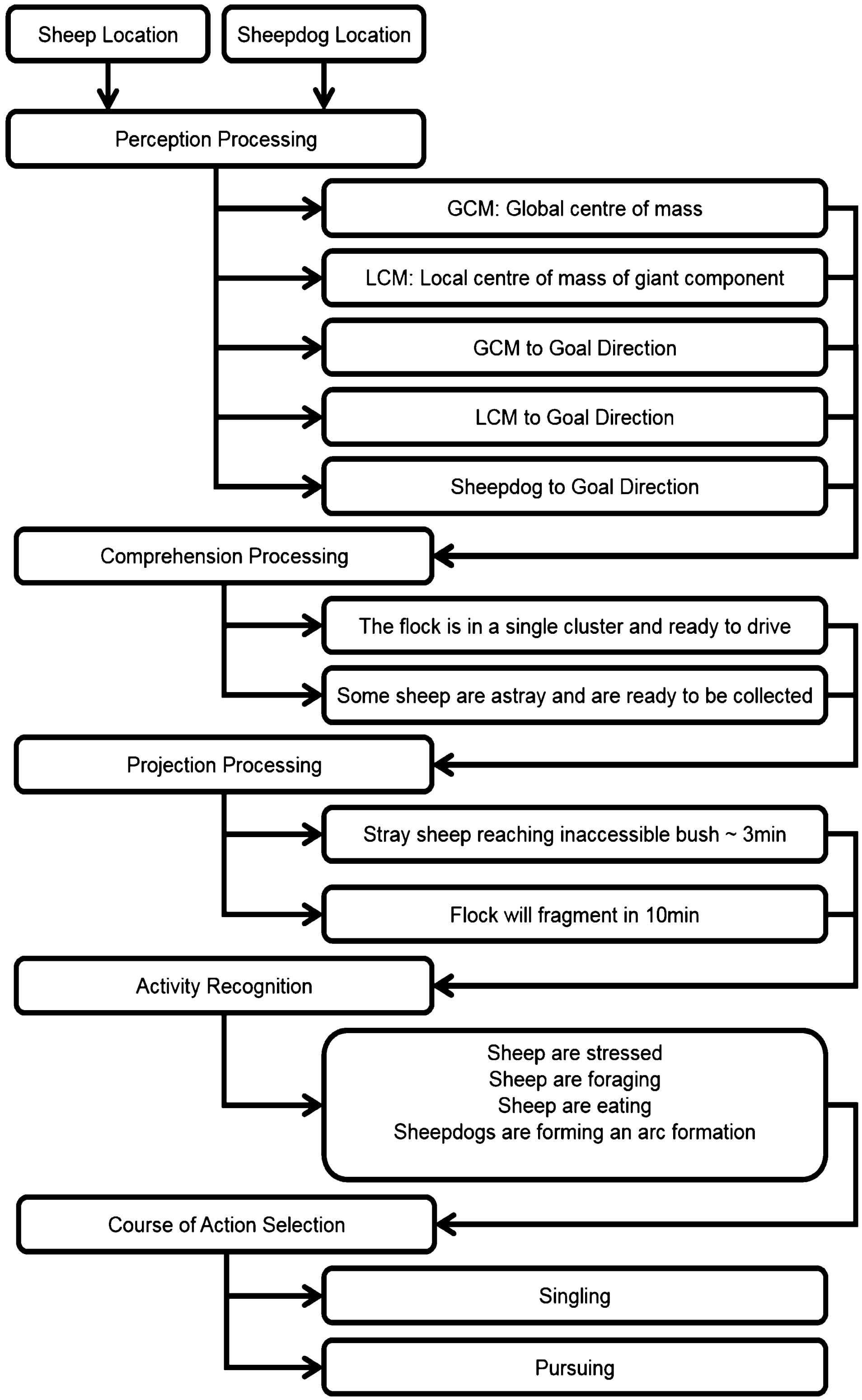

Fig. 1.5 A shepherding example using key components of the contextual awareness architecture

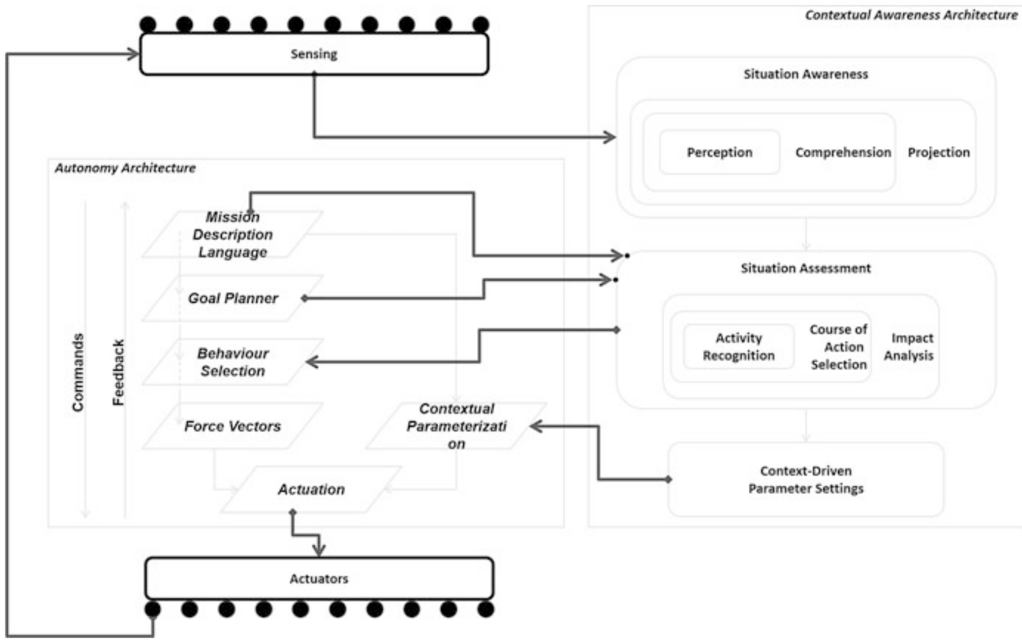

Fig. 1.6 An architecture for smart shepherds and sheepdogs, linking together the previous two sub-architectures. It empowers mission planning and decision making with the ability to perceive the environment and reason about it as well as the agent's actions. The lines for the two sub-architectures responsible for autonomy and contextual awareness are faded out to emphasise the additional links that connect the two sub-architectures

The links between some of the actuators and sensors allow for direct interaction with sensors. For example, the situation assessment may result in course of action to actively direct the sensors towards specific regions in the environment to collect missing information. Active sensing allows the agent to independently control its own orientation in the environment and the relative orientation of its sensors to its body, such as orienting a gimbal mounted camera on a robot sheepdog towards the furthest sheep location without changing the orientation of the robot itself. It also allows for dynamic configuration of its sensors, such as in the case of adjusting camera zoom in and out.

It is assumed that agent-to-agent communication is possible in order to form common contextual awareness across multiple shepherding agents. This is enabled through the architecture in Fig. 1.6 under the assumption that radio receivers may be represented as sensors and radio transmitters may be represented as actuators. The decision to communicate requires actions and actuation. Therefore, communication should not be seen in this architecture as a mere exchange of information. It is an intentional act to either improve another agent's situation awareness or to delegate the execution of a sub-goal to another agent. An agent may communicate to request information to improve its own situation awareness. In this case, communication is still an action of that agent and is adequately captured by this architecture.

In the contextual awareness architecture, the situation assessment module requires information about the mission and the sub-goal(s) the agent intends to execute to inform the three situation assessment sub-modules. The resultant situation assessment informs the module responsible for parameters' settings, which are sent to the contextual parametrisation modules in the autonomy architecture (ultimately enabling appropriately weighted and parameterised force vector aggregation). Actions are generated in the situation assessment module. These actions represent the tactics of an individual. The behavioural selection module selects the individual behaviours to achieve these tactics. The separation between tactics and individual skills in the two architectures is intentional. It allows the agent to change its skill sets available to achieve the same tactics. It also allows the agent to change its tactics by reusing its skills. In essence, tactics exist to achieve the sub-goals selected by the autonomy architecture, while behaviours are the sequence of actions to achieve a tactic.

AI methods and algorithms sit at the core of each module. Even though we propose this architecture to enable cognitive agents as a step towards realising true autonomy, a spiral design approach could see the overall architecture implemented with very simple rules in each module. For example, the rule-based system used by Bayazit et al. [1, 3], or the repulsion–attraction equations approach followed by many researchers such as Lien et al. [8, 10] Miki and Nakamura [13], Lee and Kim [9], Tsunoda et al. [23], Strömbom et al. [21], and Hoshi et al. [5, 6] could all be used to design a functional, artificial, and reactive sheepdog. In this case many of the modules in the architecture could arguably be empty, or trivial, given that these models assume flock state information is readily available and/or that the agent has global information. Such assumptions preclude the need for active sensing; the situation awareness module would be bypassed, as all required information is readily available in the agent. Similarly, the situation assessment module would be empty due to the pure reactive nature of the agent. Our architecture allows for the expansion of such approaches for implementation where overly favourable assumptions surrounding availability of knowledge are relaxed.

1.7 Conclusion

With the concept of shepherding introduced and its applicability to the many real-world problems explained, we have argued the need for cognition in order to enable effective swarm control through the use of artificial shepherds and sheepdogs. In order to improve transparency in decision making of such entities and to ease the learning burden upon AI systems in this pursuit, we have provided an ontology for shepherding. To enable implementation of smart shepherding agents we have introduced a modular architecture combining autonomy in decision making, and contextual awareness applicable to both artificial shepherds and sheepdogs. We contend that the complexity associated with implementing a smart shepherding agent lies in providing the agent with the right information at the right time. An artificial sheepdog does not need to be an exact replica of a biological one, it needs only to achieve the same objective. As such, it may not need to act on the same information sensed and used by a real sheepdog, however, the information it does obtain needs to be acquired and processed in a timely fashion given the dynamic nature of its task.

To emulate the functionality of biological sheepdog, the artificial incarnation requires AI algorithms ranging from those analysing data such as clustering, classification, and point prediction to those making plans and decisions such as Bayesian belief networks and reinforcement learning. For example, classification algorithms might be needed in the activity recognition module, clustering in the comprehension module, point prediction in the projection module, reinforcement learning in the goal planning module, and more. The course of action selection and impact analysis modules primarily rely on recommender systems with appropriate optimisation, simulation, and search algorithms.

While the behaviours of an agent here are expressed in the form of force vectors similar to those presented in Miki and Nakamura [13], the information needed as inputs to these force vectors and the parameterisation of these equations, including the selection of which behaviours to exhibit in which situation, all require AI algorithms.

We conclude that the advent of the artificial sheepdog should further enhance shepherding capability. Distributed AI algorithms in the areas of task allocation, decision making, and planning are possible upon radio equipped artificial shepherding agents and would enable optimised coordination upon the sheepdog pack; a capability alien to their biological counterparts.

The remaining chapters in this book are of various granularity. Some zoom out and discuss the complexity of a topic, such as activity recognition or human-swarm teaming, offering conceptual designs without concrete functional implementation. Others zoom in on a particular algorithm to solve a problem, such as those looking at deep learning to design controllers for shepherding or swarm decision making algorithms to disambiguate a global system-level state from local information collected by many agents. The variety of the contributions in the following chapters showcase the richness of the problem space, which is one of the reasons we are drawn to it. Clearly, implementing a fully autonomous smart sheepdog is a non-trivial system-of-systems problem, with potential application in many domains. It begs for the consolidation of different AI algorithms, and engineering modules; we hope that our architecture can help enable this and invite you on our journey.

Acknowledgement The work in this chapter is funded by the Office of Naval Research Global.

References

1. Bayazit, O.B., Lien, J.M., Amato, N.M.: Better Group Behaviors Using Rule-Based Roadmaps, pp. 95–111. Springer, Berlin (2002). https://doi.org/10.1007/978-3-540-45058-0-7
2. Bonabeau, E., Dorigo, M., Marco, D.d.R.D.F., Theraulaz, G., Théraulaz, G., et al.: Swarm Intelligence: From Natural to Artificial Systems. Oxford University Press, Oxford (1999)

3. Burchan Bayazit, O., Jyh-Ming Lien, Amato, N.M.: Roadmap-based flocking for complex environments. In: 10th Pacific Conference on Computer Graphics and Applications, 2002. Proceedings., pp. 104–113 (2002). https://doi.org/10.1109/PCCGA.2002.1167844
4. Gage, D.W.: Command control for many-robot systems. Technical Report, Naval Command Control and Ocean Surveillance Center RDT And E Div., San Diego, CA (1992)
5. Hoshi, H., Iimura, I., Nakayama, S., Moriyama, Y., Ishibashi, K.: Computer simulation based robustness comparison regarding agents' moving-speeds in two- and three-dimensional herding algorithms. In: Joint 10th International Conference on Soft Computing and Intelligent Systems (SCIS) and 19th International Symposium on Advanced Intelligent Systems (ISIS), pp. 1307–1314 (2018). https://doi.org/10.1109/SCIS-ISIS.2018.00205
6. Hoshi, H., Iimura, I., Nakayama, S., Moriyama, Y., Ishibashi, K.: Robustness of herding algorithm with a single shepherd regarding agents' moving speeds. J. Signal Process. **22**(6), 327–335 (2018). https://doi.org/10.2299/jsp.22.327
7. Hunjet, R., Stevens, T., Elliot, M., Fraser, B., George, P.: Survivable communications and autonomous delivery service a generic swarming framework enabling communications in contested environments. In: MILCOM 2017–2017 IEEE Military Communications Conference (MILCOM), pp. 788–793. IEEE, Piscataway (2017)
8. Lien, J.-M., Rodriguez, S., Malric, J., Amato, N.M.: Shepherding behaviors with multiple shepherds. In: Proceedings of the 2005 IEEE International Conference on Robotics and Automation, pp. 3402–3407 (2005). https://doi.org/10.1109/ROBOT.2005.1570636
9. Lee, W., Kim, D.: Autonomous shepherding behaviors of multiple target steering robots. Sensors **17**(12), 2729 (2017)
10. Lien, J.M., Bayazit, O.B., Sowell, R.T., Rodriguez, S., Amato, N.M.: Shepherding behaviors. In: IEEE International Conference on Robotics and Automation, vol. 4, pp. 4159–4164. Citeseer (2004)
11. Long, N.K., Sammut, K., Sgarioto, D., Garratt, M., Abbass, H.A.: A comprehensive review of shepherding as a bio-inspired swarm-robotics guidance approach. IEEE Trans. Emer. Topics Comput. Intell. **4**, 523–537 (2020)
12. Masehian, E., Royan, M.: Cooperative control of a multi robot flocking system for simultaneous object collection and shepherding. In: Computational Intelligence, pp. 97–114. Springer, Berlin (2015)
13. Miki, T., Nakamura, T.: An effective rule based shepherding algorithm by using reactive forces between individuals. Int. J. InnovativeComput. Inf. Control **3**(4), 813–823 (2007)
14. Org, I.: Cambridge dictionary. https://dictionary.cambridge.org/dictionary/english/
15. Reynolds, C.: Flocks, herds and schools: A distributed behavioral model. In: Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '87, pp. 25–34. ACM, New York (1987)
16. Reynolds, C.W.: Steering behaviors for autonomous characters. In: Game Developers Conference, vol. 1999, pp. 763–782. Citeseer (1999)
17. Schultz, A., Grefenstette, J.J., Adams, W.: Roboshepherd: learning a complex behavior. Rob. Manuf. Recent Trends Res. Appl. **6**, 763–768 (1996)
18. Singh, H., Campbell, B., Elsayed, S., Perry, A., Hunjet, R., Abbass, H.: Modulation of force vectors for effective shepherding of a swarm: A bi-objective approach. In: 2019 IEEE Congress on Evolutionary Computation (CEC), pp. 2941–2948. IEEE, Piscataway (2019). https://doi.org/10.1109/CEC.2019.8790228
19. Smith, P., Hunjet, R., Khan, A.: Swarm learning in restricted environments: An examination of semi-stochastic action selection. In: 2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV), pp. 848–855. IEEE, Piscataway (2018)
20. Spears, W.M., Spears, D.F.: Physicomimetics: Physics-Based Swarm Intelligence. Springer Science & Business Media, Berlin (2012)
21. 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. Interf. **11**(100) (2014). https://browzine.com/articles/52614503

22. Tang, J., Leu, G., Abbass, H.A.: Networking the boids is more robust against adversarial learning. IEEE Trans. Netw. Sci. Eng. **5**(2), 141–155 (2017)
23. Tsunoda, Y., Sueoka, Y., Osuka, K.: On statistical analysis for shepherd guidance system. In: 2017 IEEE International Conference on Robotics and Biomimetics (ROBIO), pp. 1246–1251 (2017). https://doi.org/10.1109/ROBIO.2017.8324588
24. Wooldridge, M.: An Introduction to MultiAgent Systems. Wiley, New York (2009). https://books.google.com.au/books?id=X3ZQ7yeDn2IC
25. Yaxley, K.J., Joiner, K.F., Bogais, J., Abbass, H.A.: Life-learning of smart autonomous systems for meaningful human-autonomy teaming. In: Handley, H., Tolk, A. (eds.) A Framework for Human System Engineering Applications and Case Studies.