Author Queries

AQ1	We have shortened the chapter title. Please confirm this is fine.
AQ2	Please cite the website address "http://www.dodccrp.org/files/NATO_COBP.pdf" in main text.

10

APPLIED COMPLEXITY SCIENCE: ENABLING EMERGENCE THROUGH HEURISTICS AND SIMULATIONS

Michael D. Norman[1], Matthew T.K. Koehler[1], and Robert Pitsko[2]

[1]*The MITRE Corporation, Bedford, MA, USA*
[2]*The MITRE Corporation, McLean, VA, USA*

INTRODUCTION – A COARSE-GRAINED LOOK

Overview

This chapter is an attempt to create a practical synthesis of complexity science and traditional engineering: introduces a set of heuristics to engineer for emergence, explores a number of them with a simulation of unmanned vehicle (UxV) swarms, and provides taxonomy to understand the levels of autonomy in engineered UxVs.

The chapter is organized into five main sections:

- *Introduction – A Coarse-Grained Look section*[1] introduces the fundamental concepts of complex systems, many of which occur in the natural world, and how they may be united with traditional engineering concepts by leveraging observations and research from the field of complexity science,

[1]In the Section 1 introduction, technically specific terms will be used without a clear definition given. The precise definitions for these terms can be found in Section 2. If starting with a detailed description of terms is important to the reader, the second section should be read first, then the introduction, which provides a coarse-grained perspective.

Emergent Behavior in Complex Systems Engineering: A Modeling and Simulation Approach, First Edition. Edited by Saurabh Mittal, Saikou Diallo, and Andreas Tolk.

AQ1

- *Definitions and a Taxonomy for Applied Complexity Science section* provides definitions for concepts such as complex systems and emergence and presents a framework for using simulation to inform systems engineering (SE),
- *Heuristics for Applying Complexity Science to Engineer for Emergence section* introduces heuristics for applying complexity science to engineer systems for emergence,
- *Unmanned Autonomous Vehicle (UxV) Swarms section* presents an unmanned vehicle swarm case as an exemplar for the integration of simulation and engineering, and
- *Operational UxV Swarms section* discusses the state of the art of engineering for emergence in the domain of UxV swarms, motivates the application of applied complexity science, and suggests a path forward.

Complex Systems and Emergence: A Path to Resilience

Why are we interested in complex systems? We are interested in them because of their remarkable abilities to adapt, evolve, and produce aggregate dynamics and/or effects that are more than simply the sum of their components' contributions or effects. These aggregate dynamics or effects are *emergent*, and they exert influences on the system and its environment that may lead to surprising system performance (positive or negative).

These aggregate dynamics or effects cannot be fully understood via examination of components in isolation. Often the emergent properties of a system will persist in a way that component interchange, adaptation, removal, or addition is of little consequence; here, we find the "ghost in the machine" (Ryle, 1949). A standing wave in a river is an example. Even though the water molecules constantly change and debris flows along with the water, the wave persists. Emergence is a fascinating topic and a unimaginably difficult problem for engineering because it is unapproachable using the traditional scientific perspective of reductionism (Kauffman, 2010). General work in applying complexity science to traditional engineering practices has been done in the fields of complex SE (Norman and Kuras, 2006; Norman, 2015) and enterprise SE (Rebovich and White, 2010). In this chapter, we try to make progress on the practical integration of complexity science and traditional engineering.

Often the concept of emergence, much like complexity, is perceived in either a positive or a negative light. The emergent and often dynamic patterns that are brought into existence can be useful, for example, the symphony of variably phased oscillations of neuronal activity that are the signature of a healthy brain. They can also be destructive, as is the case when complex networks of neurons all become entrained to oscillate in phase, creating an epileptic seizure (Jirsa *et al.*, 2014).

Not only do complex systems adapt and evolve, they do so in a self-organized manner, without any central control mechanism (Holland, 1992; Strogatz, 2004; Taleb, 2012); this is found in even one of the most complex systems, the human brain. Although cognitive neuroscientists have discovered cortical subnetworks that are relatively specialized (Tononi *et al.*, 1994), there is no homunculus instructing the brain on how to organize itself (Bar-Yam, 2002). This self-organization can come from two basic mechanisms, a self-evolutionary manner (Bak *et al*., 1987) and from external design or engineering (Carlson and Doyle, 1999).

Adaption and evolution are not typical capabilities of complicated systems. Informally defined, a complicated system is one that is composed of many components, in which each component is only connected to a relatively small number of the other components; each component's behavior is almost entirely prescriptive, and the whole system can be understood through reductionist methods (e.g., linear analysis). Complicated systems can be easily recognized because they are engineered, deployed, and employed in discrete steps using linear analytical methods (e.g., superposition in circuit analysis).

Aristotle's eloquently put, "the whole is greater than the sum of its parts" (Von Bertalanffy, 1972), has become a hallmark of emergence and a signal of underlying complexity. Dissecting a complex system to study it loses information about the relationships that existed between the components. Many readers will undoubtedly have examples of such systems readily at hand: economies, natural ecosystems, and social networks all share a set of remarkably similar properties; most of which we are still just beginning to identify and understand. A thorough discussion of the shared properties of organically formed complex systems (Holland, 1995; Bar-Yam, 1997; Wilson, 2000; Miller, 2016) is beyond the scope of this chapter.

One shared property of complex systems that is of interest to those who would like to study and design them is their potential resilience (Gao *et al*., 2016) to particular classes of perturbation and, in some cases, to antifragility (Taleb, 2012). In the sections that follow, we turn our attention to the process of designing and engineering systems that have emergent properties.

Engineering Resilience

Traditional Systems Engineering Traditional SE has little to offer in solving problems of complexity (Norman, 2015). The reader is encouraged to review the foundational SE literature (Norman and Kuras, 2006; Rebovich and White, 2010; Pitsko, 2014) relevant to the topic. Suffice to say, the SE process of gathering all requirements before any system development or field testing (Blanchard *et al*., 1990) is neither realistic nor especially helpful while designing systems that need to perform in dynamic, perturbed environments (Pitsko, 2014; Norman and Koehler, 2017).

Perturbing Complex Systems Whenever we make a change to an organic or designed complex system, we are perturbing it. Examples of perturbing complex systems include introducing, changing, or removing economic; environmental; or social policies. These perturbations can come from agents within the system, emergent effects, environmental input or outside forces that are changing the composition or "physics" of the system, or applying some sort of stress to it.

Designing Complex Systems Designing complex systems, sometimes referred to as "complex systems engineering" (Norman and Kuras, 2006; Sheard and Mostashari, 2009), is not a common human skill, but the few successes have resulted in disruptions. Technological examples of disruptive, designed complex systems include the TCP/IP stack (Norman, 2015), block chain-based technology (e.g., ethereum) (Brito and Castillo, 2013), peer-to-peer networking (Schollmeier, 2001), and Wikipedia (http://www.wikipedia.org).

A non-technological example of a designed complex system would be the United States Government as defined by the constitution. The decision by the U.S. founders to create a republic was based on the intuitive understanding of a pure democracy's emergent, self-destructive properties if executed at a large-enough scale (Bovard, 1995). Their solution for avoiding the negative emergent effects of a large-scale direct democracy was a republic of many smaller states. Thus, the founders sought to create a system that avoided the tragedy of the commons and enabled perturbations to be dealt with at a proportional scale. The founders also knew of the fragility that comes from a strict hierarchy due to its inability to deal with multi-scale complexity (Bar-Yam, 2004b). Although it may or may not be the case that the given system's designers intended to create a complex system, that is what resulted in the above examples.

Our simple goal for this chapter is to document the process of designing a set of rules, which will produce desirable emergent effects when computationally modeled. It is often the case that minimalist changes trump more interventionist ones. The next section describes the lexicon of Applied Complexity Science in explicit detail.

DEFINITIONS AND A TAXONOMY FOR APPLIED COMPLEXITY SCIENCE

Rigorous Definitions

This section provides rigorous definitions for the complexity science concepts being explored. These definitions are made explicit to properly ground the computational system design.

Complex Adaptive System

Although there is no universally agreed definition of a complex adaptive system, there are a number of commonalities that suffice for our purpose (Miller and Page, 2009; Holland, 1995):

- some number of discrete entities
- heterogeneity
- interactions
- a meaningful space
- learning/adaptation (sometimes).

So, given these components and our purpose, we define a complex (adaptive) system as a system that contains many, heterogeneous, interacting entities that exist in a meaningful space and may change their behavior over time based on their experiences. One important element of these systems is that based on the interactions among entities (positive and negative feedback) and the adaptation or change that may occur, the dynamics of the system will likely be nonlinear especially when reacting to a perturbation (Lorenz, 2000). Moreover, emergence makes these systems especially interesting.

Emergence

Emergence is often defined as the manifestation of a surprising feature of a complex system. In fact, the field of complex systems has even been characterized as the "science of surprise" (Casti, 1994). Unfortunately, the term "surprise" remains difficult to describe rigorously and remains a function of the observer. Therefore, while we will not attempt to propose a definitive definition of emergence in this chapter, we will use a working definition: an emergent phenomenon is a system-level dynamic not seen in any one, isolated component and, therefore, is not "easily" discovered, a priori, through a reductionist approach (e.g., linear superposition).

As an example, even though ants are very simple creatures, an ant colony can produce optimal foraging paths, build structures, farm, go to war, and so on (Hölldobler and Wilson, 1983). It would be difficult, if not impossible, to discover these features of the colony via a careful examination of an individual ant in isolation. It takes a number of ants, interacting within an environment, to create these colony-level features. This raises the issue of scale; how many ants are needed to create these emergent properties? Although this issue has been discussed (Anderson, 1972), it has received little systematic attention. We will discuss this in the following sections.

Often a complex system must be computationally modeled to understand the rules of interaction (and thus the agent decision space) that drive the emergent phenomenon. The advent of modern computing capabilities has enabled generative and non-reductionist approaches to modeling emergence.

It should be noted that one of the most striking features of emergent phenomena is its ability to exert influence on the system's constituent components. Consider, as an example, the myriad agents making market decisions to buy/sell a security leading to that security's dynamic market price. According to our working definition of emergence, the value of a security is an emergent entity with its own top-down influence on the agents that make up the market: dynamic price action directly influences a person's decision to buy, sell, or hold a security.

Complicated System

In contrast to complex systems, a complicated system is a sum of the constituent components, which behave in a linear manner. Here, reductionist approaches are perfectly fruitful. Moreover, the behavior of the system is predictable through a careful examination of the components and their stable combinations. A passenger car is an example of a complicated system. By examining the brake pedal, an observer can understand that it is designed to increase the pressure in the brake lines via a hydraulic system, which, correspondingly, increases the pressure of the brake pads against the rotors. Finally, this pressure causes friction dissipating kinetic energy as heat and eventually brings the vehicle to rest. Importantly, unless there is a failure of some sort, the brake pedal will only do that. The order of operations of other systems in the passenger car does not matter to the brake.

Engineering

The Academy of Engineering (www.nae.edu) stresses that engineering is the constraint-based application of science. Importantly, the academy also stresses that "to 'engineer' a product means to construct it in such a way that it will do exactly what you want it to, without any unexpected consequences" (https://www.nae.edu/About/FAQ.aspx). Clearly, this implies the engineering of linear systems, whose dynamics are predictable.

If engineering is all about the design and creation of systems that behave predictably, how then can engineering possibly make use of complexity science where a defining characteristic is that system-level dynamics are unpredictable based on an examination of components in isolation? We contend, with modern computing hardware and simulation software, one can complement existing engineering tools and methods to potentially explore the emergent properties of a system *as it is being designed*. Existing engineering tools and techniques can be used to design individual components of a system and the representation and interaction of these components can be explored with tools and techniques of complexity science to understand the corresponding emergent system-level properties.

As depicted in Figure 10.1, we see the union of the tools and techniques of engineering and complexity science as necessary for the practice of complex SE. These tools and techniques complement each other. Traditional engineering can be used to create components and then complexity science can be used to evaluate the performance of the overall system (the components in aggregate). This, in turn, may suggest changes for the individual components to improve system-level performance and the cycle begins again. Note that this process can be initiated from either the engineering or complexity science domains; system design can begin with an exploration of dynamics as a way of informing prototypical engineering efforts that seek to explicitly enable emergence.

Utility Function

It takes sufficient understanding of the overall system utility function (what does success look like) to enable and make use of emergence in complex systems. The overall system utility function is closely related to the utility functions of the individual components. As each component tries to maximize its own utility, the overall system utility function increases (for early work to create a foundation for a science of complex systems design, see (Tumer and Wolpert, 2004)). In a truly zero sum environment, this may not be possible, meaning that leveraging emergence cannot be done in all cases. As discussed by Tumer and Wolpert (2004), leveraging the emergence of

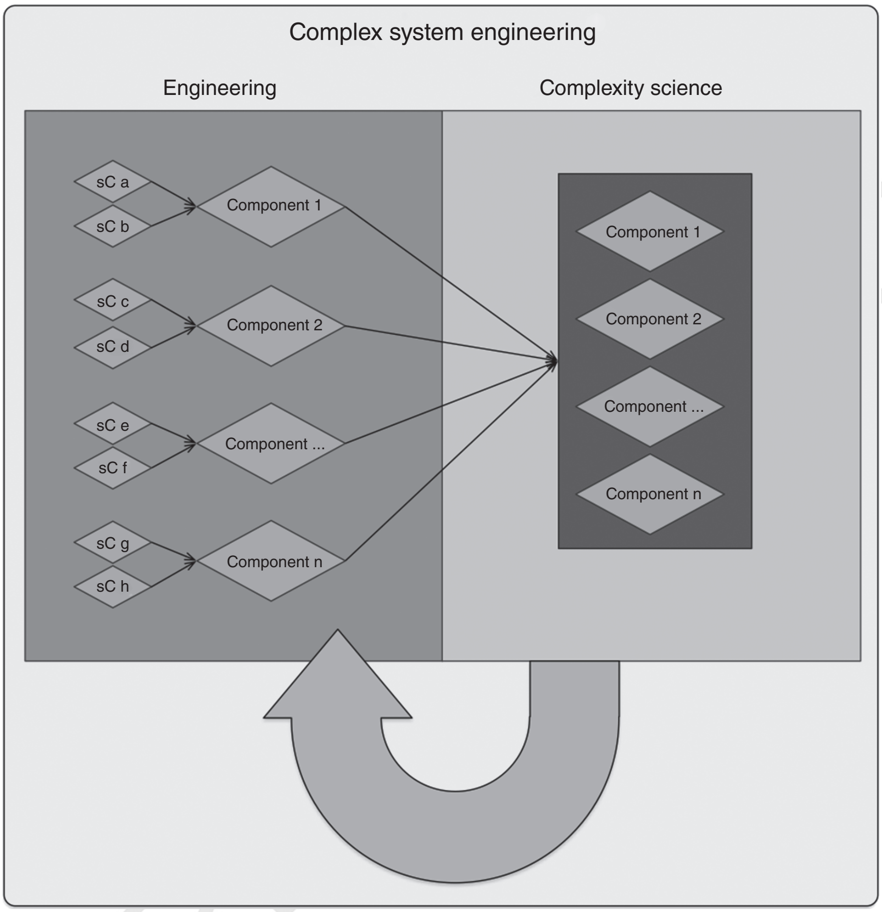


FIGURE 10.1 Complex systems engineering. The feedback between traditional engineering practice and complexity science. The emergent phenomenon exhibited at the largest scale of the system's existence (the right side of the figure) has a top-down influence on the system's components (created via traditional engineering of subcomponents (sC a–h) that are combined into larger system components (Component 1 … n)). Traditional systems engineering practices do not pay mind to this cross-scale feedback loop, the study of which is a primary focus of complexity science.

collective dynamics can be difficult and is driven in part by the scale of the system, the control within or over the system, and system-level information available to the components.

Tumer and Wolpert (2004) continue with two more important points: the collective can be designed via model-based design (meaning the components use a model of the system for decision-making) or via a model-free design (meaning the components react to their environment without an understanding of the overall system). Finally, Tumer and Wolpert (2004) characterize the human challenge as that of a "forward" problem (trying to analyze and understand existing systems) and of an "inverse" problem (where the desire is to design a complex system from scratch).

Here, we focus on the latter. Before discussing those, it is necessary to explore the decentralized system design heuristics that various pursuits in complexity science have uncovered.

HEURISTICS FOR APPLYING COMPLEXITY SCIENCE TO ENGINEER FOR EMERGENCE

Enabling Emergence

There are a number of taxonomies of complex system and emergence "types" (Maier, 1996; Bar-Yam, 2004b; Miller and Page, 2009). No matter what taxonomy one chooses to use, to date, designed complex systems fall to the less emergent and adaptive end of the spectrum compared to natural ones and represent just the beginning of a new era of design. This chapter describes a novel perspective and terminology to allow us to look further into the future of complexity and automation than we have previously been able to do. It is a synthesis of the seemingly antithetical worlds of engineering and complexity science, which we refer to as *Applied Complexity Science*.

Applied Complexity Science seeks to *engineer for emergence* in the pursuit of effectively perturbing existing systems or designing new complex systems using evidence-based methods. It should be noted that we consider making any changes an existing complex system, for better or worse, to be classified as perturbation. The methods used to these ends include computational modeling and simulation with the goal of growing antifragile complex systems in the spirit of Taleb. The hallmark of such systems is their inherent ability to not only "bounce back" from exceedingly rare or catastrophic events but to become stronger due to them. Although engineering disciplines have not yet discovered how to create developed complex systems, we propose that the most promising path is through the application of complexity science. The following heuristics are offered as design-time and implementation-time considerations to enable emergence.

Scale Matters Things that work at one scale do not necessarily work at another. Recognition of scale leads to recognition of emergence (Anderson, 1972). Often feedback loops exist between scales. Always maintain *scale discipline* when designing and modeling complex systems. Inquiry (perturbing or designing) into complex systems often requires the scale of the question being asked to be explicitly stated, as the modeling efforts must be cognizant of and resultant solution space must accommodate that scale, as well. Scale in this context is inclusive of both size and granularity of perspective. A strict interpretation of the nuances between differing definitions of scale, scope (Ryan, 2007), and complexity (Bar-Yam, 2004c) is outside the bounds of this chapter.

A system's ability to cope with changing environmental conditions while maintaining or improving its functionality must be understood at both the scale of the component and the scale of the aggregate, and the mapping between the two must be made explicit. Strict hierarchical control systems create fragile enterprises (Norman, 2015) if the complexity at lower scales is greater than the comprehension of the leaders at the highest scale (Bar-Yam, 2004a). This has the effect of severely limiting emergent behavior and the agility, adaptability, and evolution, which enable complexity. Moreover, this "control disconnect" can lead to poor decision-making based on a misunderstanding of the actual causal relationships at lower levels of the hierarchy.

Focus on Relationships and Interactions One central theme that ties man-made complex systems together is that they are created from the perspective of a macroscale system existing as a functionally emergent property of the relationships of the microscale components. A flock emerges from a group of birds because of their interactions. Similarly, an ant colony can farm, build structures, find optimal foraging paths, and go to war not because any given ant is a brilliant tactician, architect, or mathematician but rather because they interact in meaningful ways.

Decentralize by Default Centralized systems are prone to developing single points of failure. By decentralizing and creating some level of redundancy, the likelihood that a system fails as the result of an attack, a natural hazard, or simply by bad luck affecting the central single point of failure is significantly reduced, so the system becomes more stable. Natural complex systems are fundamentally decentralized. It is important to note the distinction between a distributed system and a decentralized system. Simply put, a decentralized system is one in which each of the major components is under individual control; in a distributed system, this in not necessarily the case. When designing complex systems, we want to decentralize as much as is reasonably possible. Any of the complex systems previously discussed can lose virtually any single component and continue to function because the components can interact with themselves and their environment, making decisions at their scale of interaction using locally available information. If a poor decision is made, the consequences of that decision do not propagate (typically) through the system unless other components (which may henceforth be referred to as *agents*) (Holland, 2006) decide to continue its propagation. If, however, decision-making across components is centralized, one bad decision can cause a systemic failure (Taleb, 2012). In this way, complex systems are resilient to perturbation precisely because they are decentralized.

Cascading failures in complex systems can be avoided through decentralization; consider the prosumer's (a producer and consumer of a given property) act of producing electricity via solar panels to protect against the extreme complications built into the architecture of the power grid (i.e., centralized production and many hierarchical levels of distribution). The prosumer is now able to function without the rest of the system if those complications tip into complexity and lead to cascading failures and emergent blackouts. An electric grid where every consumer is also a producer creates a basic complex system that has an emergent property *of* electricity.

Allow Ambiguity The only way to gain traction in the midst of complexity, especially when pursuing modeling efforts, is to take a coarse-grained view of the system, allowing fine-grained details to go unrealized unless, or until, they are required for capturing the emergent properties of interest (Barry *et al*., 2009). The search for the proper level of abstraction that captures enough details of the system's relationships to reproduce the emergent properties of interest is, in itself, an art form.

As shown in "Reference modelling in support of M&S—foundations and applications," (Tolk *et al*., 2013), for nontrivial systems, the incompleteness theorem is applicable, which means that consistent formal representations are necessarily incomplete, and complete representations are inconsistent. The recommended solution is a complete, but inconsistent, reference model that is used to derive a set of incomplete but consistent (simulation) systems. They form a set of orchestrated alternatives that allow focus on diverse facets and, like the hurricane map in the weather forecasts, present a whole picture based on the set of ambiguous models. Ambiguity allows one to consistently cover all known aspects of a system completely.

Like the "NATO Code of Best Practice for C2 Assessment/Selecting an Orchestrated Set of Tools" observes already (p. 199):

> The natural tendency of an analyst is to simplify a problem. Part of that simplification is to select a tool, preferably only one, which will meet the analysis requirements. In the analysis of [Command and Control (C2)] combat operations, this may be possible if the analysis is properly scoped. In the analysis of [Operations Other Than War] C2, the issues are typically too numerous, the variables are too confounding and the scope is too broad for one tool to satisfy all analysis requirements. An orchestrated set of complementary tools will normally be required.

Decide Locally Using a Model-Free Design To enable emergence, components need to make decisions without a model of the entire system; we refer to this as model-free design. This means that data that is available in the immediate environmental context is all that is required for an agent to make decisions.

More data means more aggregation time. If data must be transmitted to and then processed in a centralized location, where results of processing then have to be retransmitted to be made actionable, value is lost due to the delays across transmission, aggregation, and action. When thinking about data usage in a designed complex system, decentralized data fusion (Durrant-Whyte *et al*., 1990), in which data is stored and accessed in a completely federated manner, may be required. Furthermore, increased data may simply result in more noise, more non-constructive redundancy and either increased cognitive load on an operator or agent or further layering of machine-based assumptions and may result in making all parts worse off (Tumer and Wolpert, 2000).

Incorporate Dynamic Modeling (As Part of the Complex System Design Process) The system itself should be designed in an executable modeling environment, so that the effects of decentralized decision-making, changing environmental conditions, nested feedback loops, and other dynamics that display hysteresis and path dependence will become apparent and may be analyzed. The modeling effort is not done to make predictions about the future, but to build insights into behavior and performance, as well as to actively probe the fragility of the system under design.

Certain situations may even require the use of active modeling at run-time within the complex system itself; consider as biological analog to this the emergent complexity of a human brain anticipating a social event.

Understand Emergent Properties (i.e., Resilience) Resilience is often described as the ability of a system to bounce back from a shock (Zolli and Healy, 2013). Although this could be an apt description of the aggregate or emergent behavior we see as a complex system interacts with its environment, it is not a useful description if the goal is to design a resilient system. As we begin to prepare tomorrow's systems engineers to directly grapple with the inevitable emergent properties of the systems they create, they may be able to leverage emergent properties to acquire functionality that would otherwise be unavailable. These properties may include inter alia, resilience/robustness to certain classes of stressor/perturbation, and antifragility.

Complexity Is Not Always the Answer

To be clear, if a problem space is of a limited scale, that is, occasional system-wide failures are acceptable, then a solution does not have to take the form of a complex system. A complication-based solution may be far more cost effective given global resource-scarcity conditions and is acceptable when a failure in the system would not propagate any further into the complex network in which the system is already situated so that the emergent properties of the aggregate network are not compromised. Hierarchical centralization of control in a complex system is perfectly suitable so long as it is nested at the scale of an expendable component or set of components (Norman, 2015).

A Starting Point

No matter where a system's boundaries are drawn, the starting point for designing a system that functions without centralized control is not clear. What is clear is the need to capture the relationships of consequence to the questions being explored by modeling the structure and dynamics. We are now nearing the edge of the possible, and the rest of this chapter will explore one method of using simulation to leverage emergence, the essence of designing a complex system by applying complexity science. Table 10.1 represents an attempt to create a complexity-based taxonomy of autonomous systems to aid in further discussion.

The first column of Table 10.1 indicates the level of autonomy. The second column explains the properties a system must possess to achieve that given level. The third column lists the Applied Complexity Science design heuristics that would need to be employed in the creation of a decentralized system capable of achieving that level of autonomy.

In the examples that follow, we focus on the design of unmanned vehicle swarms (UxVs). We assume that the engineering of a specific component of Figure 10.1 is successful. The focus of this chapter is on the use of the tools and techniques of complexity science and the corresponding feedback into the traditional engineering

TABLE 10.1 A Taxonomy of Applied Complexity Science

Level of Autonomy	Properties of System	Heuristics Employed in System Design
0	• Micromanagement of systems • Distributed design ■ Preprogrammed travel paths	None
1	• Emergence levered via local decision-making for navigation of agents in system space ■ Tactical goals established externally	1–3
2	• L1 + decentralized design at large scale (including decentralized decision-making using local information)	1–7
3	• L2 + agents have limited ability to vary parameters of their own rules of interaction • May be based on competition of internal agent models to guide *adaptation* (self-induced change of the agent)	1–7
4	• Antifragile complex systems capable of designing themselves • L3 + agents have ability to change themselves and propagate/fuse those changes to/with others • Rate of system *evolution* (self-induced change of a system-wide/emergent property) may be artificially accelerated (by either internal or external influence); "policy vote" ■ Altering aggregate goals in response to environmental perturbations • Generate a collective solution when confronting a problem at a collective scale; that is, *emergent computation* • Future swarm strategies are formed and tested via emergent computation across the agent base	1–5, 6 (applied recursively), 7

process that may occur. We also assume that the system must be able to scale and that we have little control over the environment. This means that each member of the UxVs will not be able to have a model of the larger system available for local decision-making. Therefore, the UxVs will need to be designed in a model-free way, with the swarm components reacting to each other and their environment without global knowledge and without the ability to communicate with all other members of their swarm. This presents a challenge, as we must create a set of rules and incentives for the members of the swarm so that as selfish maximizers, they will create a high-performance collective capable of emergence.

UNMANNED AUTONOMOUS VEHICLE (UxV) SWARMS

The example complex system designed in this chapter is a swarm of autonomous unmanned vehicles (UxVs). The agent-based modeling platform NetLogo 6.0.1 is used for this purpose. Although we use the UxV example to explore many of the ideas discussed above, we cannot cover them all. The examples in this chapter explore the following heuristics of Applied Complexity Science: scale matters, focus on the relationships, decentralize by default, allow ambiguity, use dynamic modeling, understand emergent properties, and decide locally using model-free design.

The future operational environment will make use of increasingly large numbers of increasingly autonomous entities. As scale increases, it is not possible for a human to command each entity directly. With the highly prototypical Perdix effort, the Defense Advanced Research Projects Agency (DARPA) has signaled the research community's inability to scale manpower across a swarm and the desire to enable swarm autonomy (Perdix Fact Sheet). Available artifacts around the Perdix program dated as recently as January 2017 indirectly claim that the system exhibits two emergent properties: the ability to adapt to some amount of swarm population dynamics and the ability for swarm agents to navigate without prespecified paths but with individually prespecified goal locations.

Based on qualitative analysis of publicly available videos of recent tests of the Perdix swarm and associated command and control, we believe that the Perdix program achieves autonomy somewhere between levels 0 and 1 on the Taxonomy of Autonomy in Applied Complexity Science, as presented in Table 10.1. Any further specifics on higher levels of autonomy planned for the Perdix program are currently unknown.

As a one-to-many mapping of operator-to-swarm agents seems to be the direction of current research interest, one of our goals is to discover what that mapping looks like and begin to ask systematic and informed questions around operational teaming.

Emergence in Swarms - Why Is a Swarm Resilient?

There exists a tendency to treat a swarm, which is decentralized, as a distributed system. A swarm is decentralized because its components possess agency. We define agency to mean the capacity for an actor to act in a given environment, and implicitly an agent is endowed with certain degrees of freedom and decision-making responsibilities within them. At scale, this agency drives emergent properties that transcend properties of individual membership. This means the swarm is resilient to almost any localized perturbation. This is brought about by the fact that each agent is operating against only a local data picture (i.e., it is a model-free system), accepting (or oblivious to) the continuous uncertainty of nonlocal events; this creates space for emergence to be leveraged and avoids the problem of a system that can fall out of synchronization.

One important emergent property of swarms is resilience: disruptions at the local level do not lead to disruptions at the aggregate level. If the tipping point for swarm behavior has been reached (e.g., density of agents), the entities will begin to interact and make increasingly autonomous decisions about mission execution and these emergent properties of the swarm will persist and demonstrate path dependence.

To design a swarm that can demonstrate useful emergent properties, we start by determining how many swarm agents are required and what their relative density must be for emergent properties to be observed.

Simulating Emergence and Exploring Applied Complexity Science Heuristics

The design of individual UxVs is generally understood and consistent with current system engineering tools and methods. The design of swarms of UxVs, on the other hand, is less generally understood. This section focuses on developing the understanding required to begin to engineer how UxVs interact to enable emergence and set conditions for the swarm to be resilient to perturbations. The Applied Complexity Science heuristics are noted in bold font as this example is developed.

Modeling a swarm in NetLogo (*incorporate dynamic modeling*) is an exercise in *allowing ambiguity* because we are abstracting the engineering specifics of the component and resultant system to get our hands around the relationships and consequent dynamics (e.g., a radio connection between two devices may be modeled as a logical link so long as the questions being asked of the system do not pertain to spectrum management).

In the initial simulation, we test a set of canonical relationships (*focus on the relationships*) used to create a "boid flock" (Reynolds, 1987). This was used as a starting point because of the emergent pattern formation that occurs as a result of these simple rules. This minimalist approach to modeling collective behavior has produced dynamics with remarkably similar characteristics to the patterns displayed by natural flocks of birds, so it is assumed that the essence of the emergent flocking behavior has been captured, despite the intentional absence of other bird-specific details that are not believed to have an impact on the birds' relationships to one another.

We construct our model around these relationship rules: cohesion (the tendency of the UxVs to cluster together), separation (the tendency of the UxVs to not get too close together), and alignment (the tendency for the UxVs to move in the same direction). Each agent makes decisions for itself with information only about its immediate environment (*decide locally*).

The flocking behavior that the model produces is a result of completely decentralized and model-free sensing and decision-making (*decentralize by default*); the flock is an emergent entity and as such should be resilient to particular classes of perturbation (*understand emergent properties*) such as the removal of any given UxV.

Swarm coherence is defined here to be represented by the variance of headings across the group; it is assumed that this is a dynamic metric whose value will vary over time. A high swarm coherence would be represented by a low variance and a low swarm coherence by a high variance. It should be noted that a swarm's variance should never be too close to zero, as the dynamics of a swarm would be destroyed in such a situation, given the fluid nature of swarm behavior within a finite environment.

Using these rules, we then explore how the density of UxVs impacts the creation of a coherent swarm (defined as most swarm elements moving in the same direction). One UxV does not make a swarm (*scale matters*). On the other hand, if there are too many UxVs, then it will be too hard for them to find a common direction and not get in each other's way. For this experiment, we held the number of UxVs constant at 250 and varied the size of the landscape from 31×31 patches to 101×101 patches (here a "patch" is a square unit of terrain). We find that swarms form very quickly at a density of 250 UxVs in a 51×51 patch size environment. As the size of the environment is increased, it takes the swarm longer and longer to cohere. As would be expected, this general trend is seen across all runs for a given environment size, and overlap may exist when comparing performance of two individual runs from slightly different-sized environments.

Figure 10.2 depicts the resulting swarm dynamics in a collective of 250 agents as the region size is increased. This baseline helps us understand the potential impacts of design decisions. Figure 10.2 shows how changes in the density of UxV impact the ability of the UxVs to form a coherent swarm. Intuitively, we suspect that denser groups will produce emergent swarms more quickly than sparser groups. Here, we define a coherent swarm when the variance of UxV headings is very low or very high (high variance is an artifact created when the swarm is heading north, causing headings to fluctuate between 0 and 360 degrees). In all the diagrams below, the UxVs exist in a toroidal landscape that varies from 31×31 patches to 101×101 patches in 10 patch increments. Heading variances below the gray horizontal line indicate a coherent swarm.

What happens when we perturb the system? Building on the UxVs example, we use the simulation to understand the impact of perturbations on swarm performance. For example, what happens if there are obstacles in the swarm's environment? We now know that a swarm forms very quickly at a density of 250 UxVs in a 51×51 patch. Using that density, what happens as we increase the density of obstacles? At what point does the swarm break down?

After understanding the engineering constraints revolving around building a swarm in isolation, an exploration of the swarm in a representative environment is performed by adding obstacles. As we add obstacles that take up 1.1% of the space, it very quickly begins to destroy the swarming dynamics. By about 5%, the swarm headings are largely random. Very low variance indicates a coherent swarm.

If we envision a military application of this swarm, it is not difficult to imagine that 5% of the airspace in which the swarm would be operating could be restricted (airports, civilian centers, other governmental facilities, etc.). This being the case, it is worth noting that the current set of rules may not create a coherent swarm under those circumstances. However, more careful analysis of the potential obstacles should be undertaken.

In the first few simulations, obstacles are placed randomly within the landscape. We find that variance tends to increase with the number of obstacles. Highly coherent swarms are not found if four or more obstacles are introduced, as is indicated by the horizontal gray line in Figure 10.3.

In Figure 10.4, we see the impact of obstacles that are highly organized. Having organized obstacles (i.e., obstacles placed in a line) significantly disrupts the coherence of the swarm more quickly than do the unorganized obstacles (i.e., obstacles placed randomly within the environment). However, unexpectedly, as the organized object grows in size, new structures develop in the swarm dynamics. Looking at obstacle sizes 2-7, there is very little structure in the heading plots (Figure 10.4, c3–h3). However, as the size of the obstacle becomes large enough to effectively create a barrier within the swarm's area of operations, new structures emerge within the swarm dynamics as indicated by the exaggerated vertical components of both the variance and the heading plots. This is an emergent effect of the swarm avoiding the large obstacles and turning to move into free space.

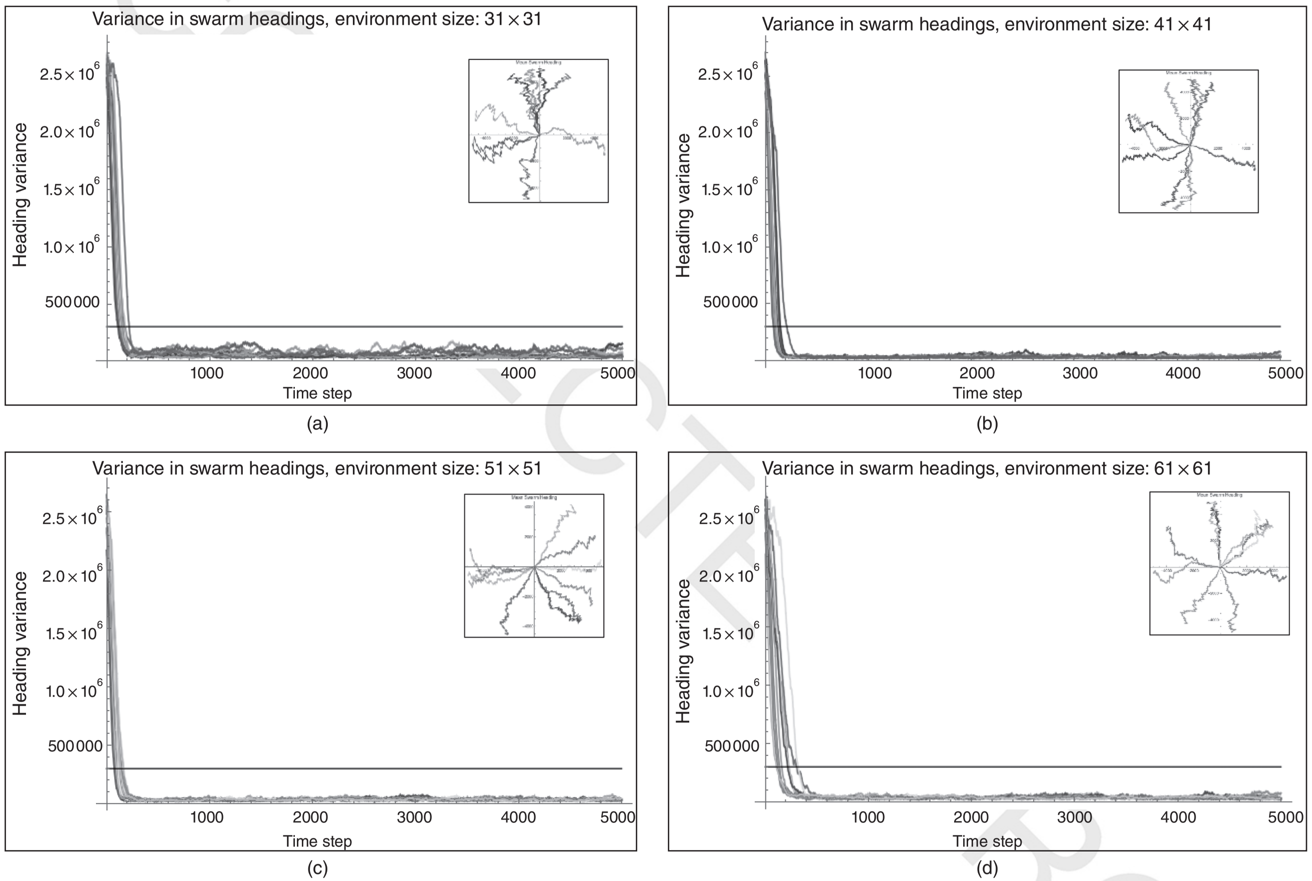


FIGURE 10.2 Sample swarm dynamics. Variance in heading of the U × V swarm as density is decreased by holding swarm size constant and varying environment from a 31 × 31 patch size to a 101 × 101 patch size in increments of 10. Swarms form very quickly at a density of 250 U × Vs in a 51 × 51 size patch. (a) Swarm heading dynamics, 250 U × Vs, 31 × 31 space. (b) Swarm heading dynamics, 250 U × Vs, 41 × 41 space. (c) Swarm heading dynamics, 250 U × Vs, 51 × 51 space. (d) Swarm heading dynamics, 250 U × Vs, 61 × 61 space.

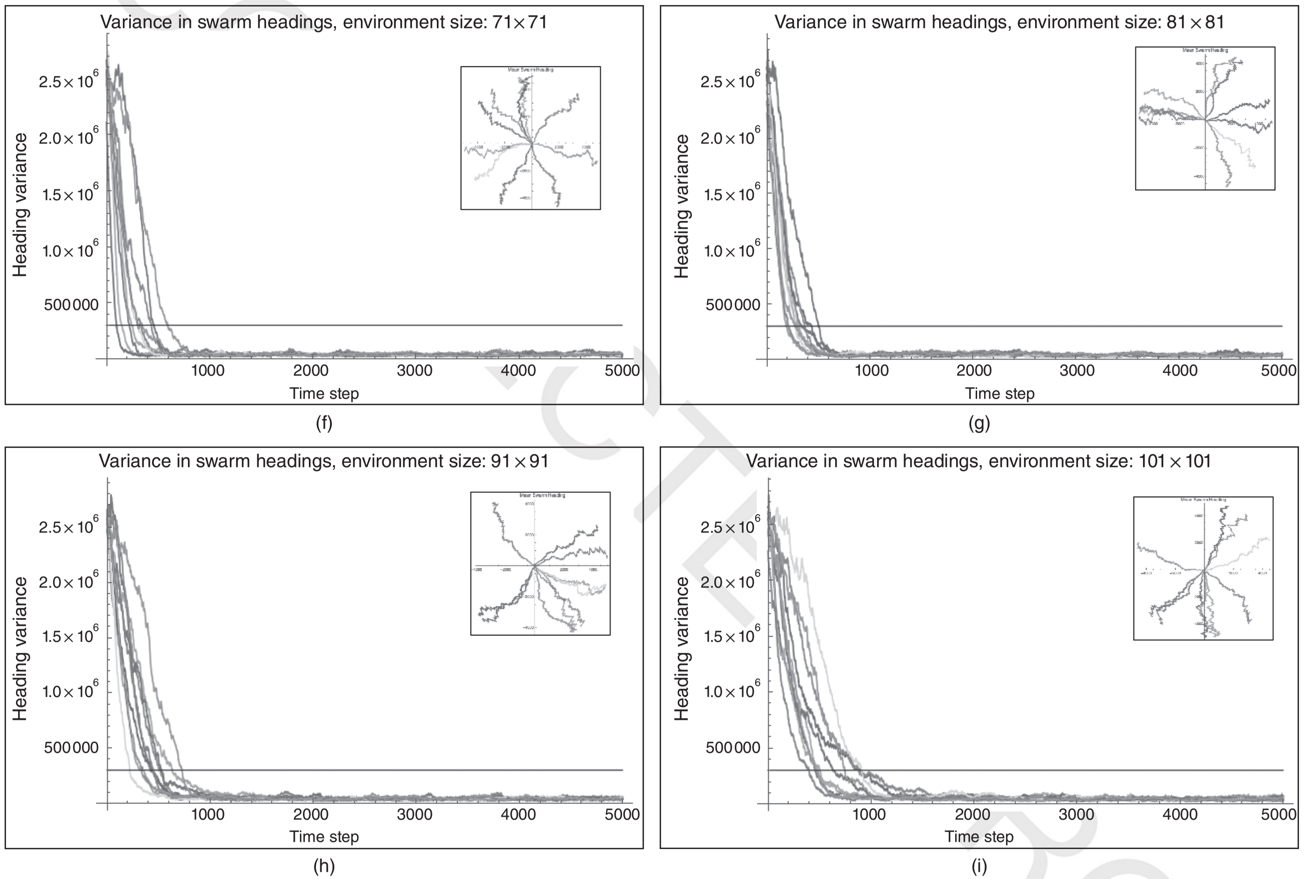
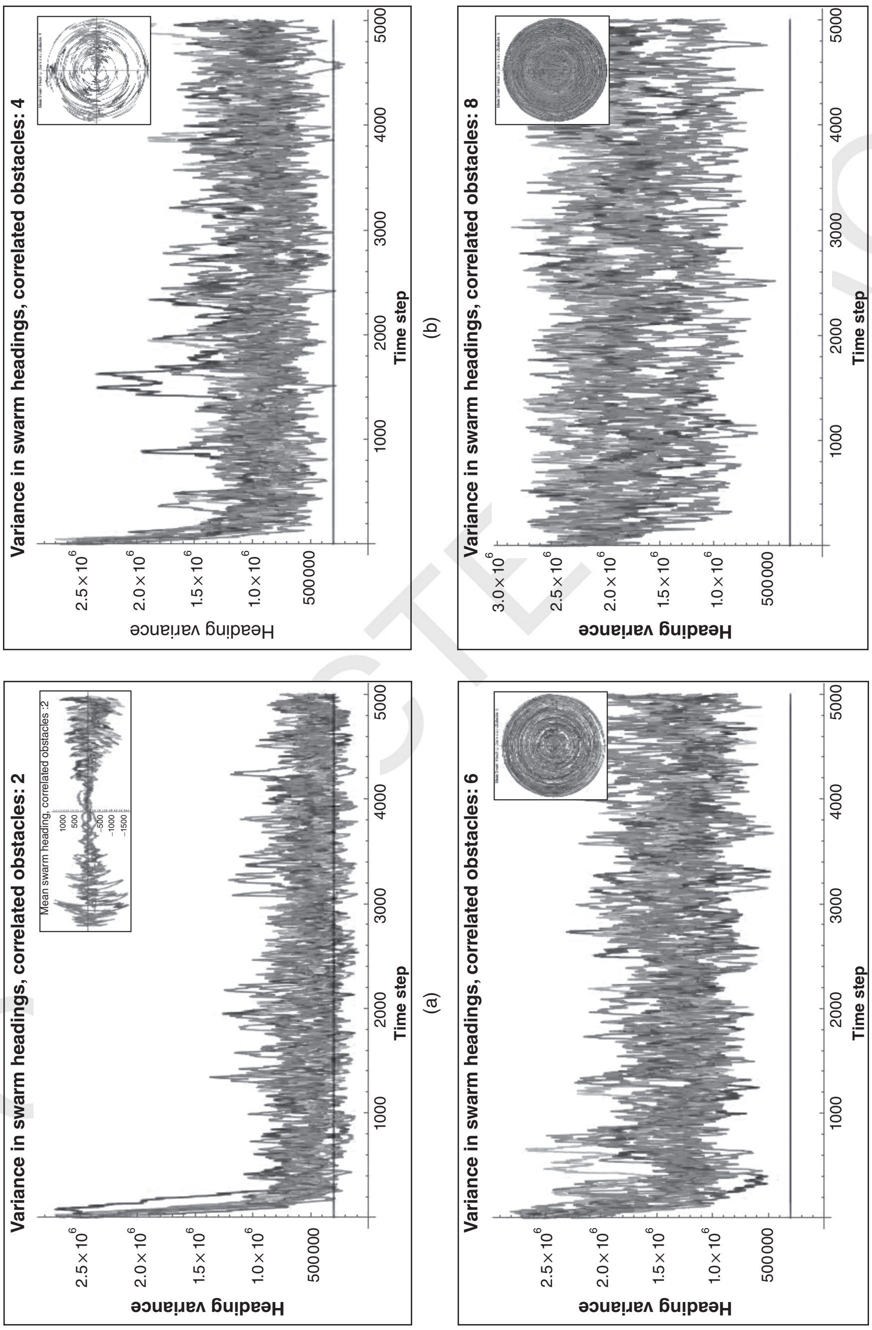


FIGURE 10.2 (*Continued*) (e) Swarm heading dynamics, 250 U × Vs, 71 × 71 space. (f) Swarm heading dynamics, 250 U × Vs, 81 × 81 space. (g) Swarm heading dynamics, 250 U × Vs, 91 × 91 space. (h) Swarm heading dynamics, 250 U × Vs, 101 × 101 space.


Variance in swarm headings, correlated obstacles: 2
Mean swarm heading, correlated obstacles :2
Heading variance
2.5×10^6
2.0×10^6
1.5×10^6
1.0×10^6
500000
1000
2000
3000
4000
5000
Time step
(a)
Variance in swarm headings, correlated obstacles: 4
Heading variance
Time step
(b)
Variance in swarm headings, correlated obstacles: 6
Heading variance
Time step
(c)
Variance in swarm headings, correlated obstacles: 8
3.0×10^6
Heading variance
Time step
(d)

What is interesting from this exploration of swarming and obstacles is that different "types" of obstacles impact the swarm differently at different scales. Unorganized obstacles act like noise monotonically causing the coherence of the swarm to break down. Organized obstacles, on the other hand, more quickly cause degradation in swarm coherence, but as their scale continues to grow the obstacles, create new coherence within the swarm.

By way of a final experiment, suppose the goal is to follow a moving object and have the UxVs spread out around it. This might be the case if there was a need to create a mesh network around a set of ground forces to relay communications in a very cluttered environment. An engineer might first try biasing the swarm by first having them all set their heading toward the ground unit and then run the swarm algorithm. Unfortunately, this rule is too strong and collapses the swarm, at best creating a line of UxVs trailing the ground units. To fix this, a rule could be added to the UxVs to spread out so that they are no closer to each other than some fixed amount. Unfortunately, such a rule does not preclude UxVs from leaving the area, thus limiting the potential of the desired UxV-based communications network. An obvious fix, therefore, is to add another rule stating that each agent should endeavor to keep at least one UxV in communications range. However, once simulated, it becomes clear that this "fix" does not preclude pairs of UxVs from leaving the group together.

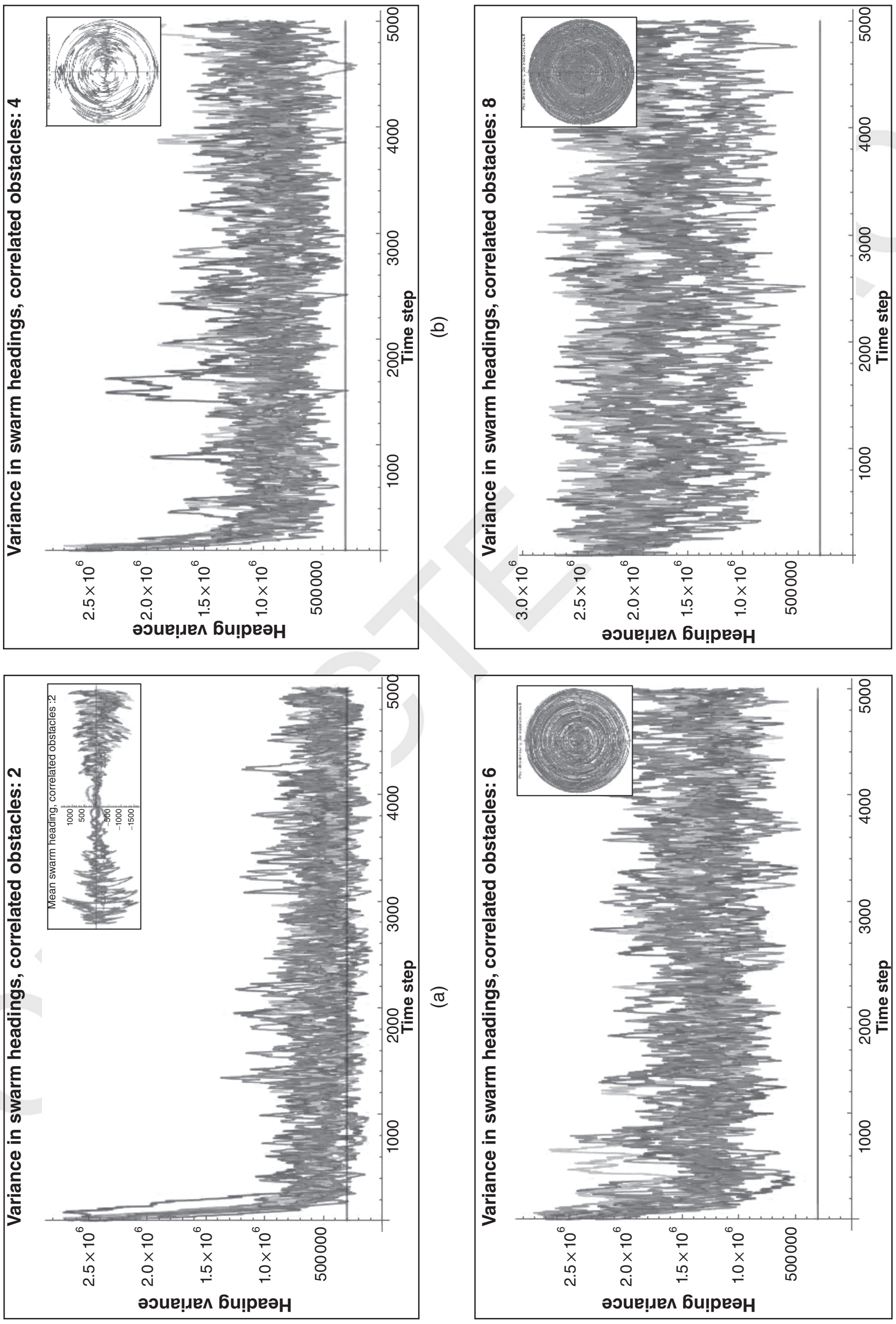
The above solutions are examples of how subtle micromanagement of an emergent phenomenon can add fragility to the system. Rather than adding to the swarm's rules, fine-tune adjustments to the existing rules may be able to be made that engender desired emergent behavior while not adding unforeseen fragility to the system.

Assuming that the UxVs move faster (in these simulations, they are required to move approximately four times, or more, faster) than the forces they are following, then the current swarm rules are usable.

First, the direction of the ground forces is included as part of the swarm heading. Second, the minimum separation distance is set to be quite high (this, in turn, may require changes to the sensor platform of the UxVs as they must be able to "see" farther than their separation distance). Finally, the maximum turn the swarm can make is set to align to roughly double that of cohere and separate turns. This generates a robust swarm that follows the ground forces and does a "good" job of spreading out around them to create a mesh network. It does not, however, guarantee coverage always as the swarm may overshoot the ground units and need to circle back.

By creating this swarm ruleset, emergence is leveraged to engineer a complex system.

FIGURE 10.3 Highly coherent swarms. The impact of unorganized obstacles on swarm dynamics, as can be seen, variance tends to increase with the number of obstacles. Highly coherent swarms are not found if 4 or more obstacles are introduced, as is indicated by the horizontal gray line. (a) Variance of heading, 250 U × Vs in a 51 × 51 world, correlated obstacle of size 2. (b) Variance of heading, 250 U × Vs in a 51 × 51 world, correlated obstacle of size 4. (c) Variance of heading, 250 U × Vs in a 51 × 51 world, correlated obstacle of size 6. (d) Variance of heading, 250 U × Vs in a 51 × 51 world, correlated obstacle of size 8.


Variance in swarm headings, correlated obstacles: 2
Mean swarm heading, correlated obstacles :2
Heading variance
Time step
(a)
Variance in swarm headings, correlated obstacles: 4
Heading variance
Time step
(b)
Variance in swarm headings, correlated obstacles: 6
Heading variance
Time step
(c)
Variance in swarm headings, correlated obstacles: 8
Heading variance
Time step
(d)

OPERATIONAL UxV SWARMS

Near-operational military swarming capabilities in the true sense of the term are nonexistent. Experimental military functions are limited to centrally controlled or distributed configurations that are micromanaged by one or more human operators or limited in scale to tens of vehicles (Wolf *et al.*, 2017). These distributed systems assume lossless transmission, complete information, shared decision spaces (maintaining a shared decision space is an incredible burden), and limitless power supplies; none of which are available in the real world. As such, emergence has not been *designed* into any operational UxV teams, although one prototypical research effort, Perdix, may be working toward that goal.

Other governmental efforts to create "swarms" have been distributed systems of very small scale, not decentralized swarms as presented in this chapter. If we were to design a system from the perspective of traditional engineering a distributed system, we would want to put identical decision-making software on each agent and give each agent access to the identical sensor/data set for continuity of decision-making, with communication channels between all agents, carrying data about everything known by all of the agents. Each agent would require this global data picture for the system to function as intended, and real-world complications such as time delay and environmental perturbations expose the fragility of a distributed system design that does not enable emergence. In a system such as this, almost any perturbation could cause the unified situational picture to become corrupted and individual agent perspectives to drift, creating divergence in the group's planning due to wrongly perceived differences in the environment. Relying on a unified global data picture is a fragile design decision. Government command and control architectures were not designed with emergent swarm behavior in mind.

DARPA's Perdix effort's contribution to leveraging emergence, if descriptions are accurate, is summarized as follows:

> Controlling 100 drones individually would be overwhelming, so much like a sport coach, operators call "plays" (e.g., surveilling a field) and Perdix decides how best to run them. Because Perdix cannot change their plays, operators can predict the swarm's behavior without having to micromanage it.

Bees are believed to participate in emergent computation during colony migration planning events (Makinson and Beekman, 2014) to guide their aggregate behavior.

FIGURE 10.4 Highly organized obstacles. The impact of organized obstacles on swarm dynamics. We can see that as the organized obstacles become large enough, new emergent swarm dynamics form within the system's new boundaries. (a) Variance of heading, 250 U × Vs in a 51 × 51 world, correlated obstacle of size 2. (b) Variance of heading, 250 U × Vs in a 51 × 51 world, correlated obstacle of size 4. (c) Variance of heading, 250 U × Vs in a 51 × 51 world, correlated obstacle of size 6. (d) Variance of heading, 250 U × Vs in a 51 × 51 world, correlated obstacle of size 8.

Expert opinion based on the available information is that the Perdix do not decide where to navigate because of decentralized design and/or emergent computation, but are given specific coordinates and specific operational objectives as part of their play-calling. This means the next clear goal for Applied Complexity Science in this space will be to increase swarm autonomy to a solid level 1, and then to level 2, as shown in Figure 10.1.

DISCUSSION AND CONCLUSIONS

Modern computing hardware and simulation software can complement existing engineering tools and methods and, potentially, be used to explore the emergent properties of a system *as it is being designed*.

Autonomous swarms must be imbued with the agency to make decisions in a decentralized manner to truly be autonomous and leverage the problem-solving power of emergent behavior, as well as perhaps glean the benefits of resilience and antifragility.

As scale increases, a complex system inevitably goes through transitions of both structure and function. It is difficult to believe that a swarm of two hundred and fifty will behave the same as a swarm of one million. Consider as an analogy the government of Switzerland. It is speculated that Switzerland's decentralized, direct democracy is what has allowed it to remain so stable while still existing at the scale of a nation-state, which has historically not been the most stable (Taleb, 2012). The Swiss system is empowered to make highly autonomous decisions at multiple scales due to Swiss Canton sovereignty. As a result, perturbations are handled at their own scale. The system adapts to the perturbations locally and prevents perturbations from cascading through the system. Therefore, the systemic fragility created by a strong centralized control mechanism never arises. An extremely large swarm should be something like Switzerland. Dynamic modeling is required to begin to implement such a system. Currently, there are no true swarm capabilities being researched or implemented by any governments or militaries we are aware of.

To fully enable and leverage emergence for swarms, one of the requirements is the need to transition from a distributed design to a decentralized design as a default (level 2 of Table 10.1). Distributed design does not necessarily involve more than one control entity, whereas decentralized design necessitates that decisions be made based only on locally available data, which means the perspective of one agent should not be identical to the perspective of another. The emergent coherence of the swarm is a result of myriad decisions made at the scale of each agent, the complexity of which becomes difficult to understand completely.

For the sake of discussion, swarm coherence is defined here to be represented by the variance of headings across the group; it is assumed that this is a dynamic metric whose value will vary over time. A high swarm coherence would be represented by a low variance and a low swarm coherence by a high variance. It should be noted that a swarm's variance should never be too close to zero, as the dynamics of a swarm would be destroyed in such a situation, given the fluid nature of swarm behavior within a finite environment.

TABLE 10.2 Swarm Coherence as a Function of Density for Environment Sizes 30×30 to 100×100

Environment Size	Minimum Heading Variance (Average Over 10 Runs)	Earliest Time Step (t) for Minimum Heading Variance (Average Over 10 Runs)
31×31	21 289.05	3183.6
41×41	18 719.23436	2338.3
51×51	17 792.29474	2697.6
61×61	18 808.29094	2565.6
71×71	19 366.58371	2914.7
81×81	20 670.87091	3850.5
91×91	21 400.06905	3082.2
101×101	22 387.4731	3853.3

The first experiment involves setting the basic foundations for exploring the swarm coherence dynamics of the canonical boids implementation. Here, the focus is on discovering the impact of swarm density on swarm coherence, without introducing internal obstacles/perturbations. From Table 10.2, it is clear that the size of the environment has a huge impact on swarm coherence. With an environment size of 100 patches, there is no convergence to a globally coherent pattern. As the swarm's density is decreased by holding the number of agents constant at 256, but the size of the environment is increased to 400 patches square, there is an unusual pattern that seems to be indicative of phase transition in swarm dynamics. Realistically, this is interpreted to be just enough space for some swarming but with enough other agents in the way that the "separate" function adds a lot of noise. As the size of the environment is increased to 30×30, it is clear that the change in swarm density is causing another phase transition in the system's dynamics to occur. For the first time, we see coherence across all experimental runs with these parameters. Coherence is specifically defined here as a swarm heading variance less than 250 000, and it is shown in Figure 10.2 as a horizontal gray line. This trend of convergence tightens until it peaks in a 50×50 environment, where the heading coherence of the entire 250 agent swarm dips below 18 000 in only 2697.6 time steps. It could also be argued that the 40×40 patch set of runs produced a highly performant swarm due to the incredibly tight variance of 18.7 K achieved in only 2338.3 time steps. As environment size grows from 60×60 onward, swarm convergence time and variance increase almost monotonically.

The major takeaway is the evidence that a "sweet spot" for swarm density exists for achieving rapid swarm coherence. High-density swarms do not function well, and low-density swarms take much longer to form.

As noted earlier, this "sweet spot" changes based on the types of environmental perturbations present. Future work will look to enable adaptation (level 3 of Table 10.1) of the agents so that goal swarm coherence metrics can be achieved across varied environmental conditions.

The system demonstrates remarkable resilience to perturbation because it is *decentralized*. This experimentation demonstrates that swarm coherence is destroyed when all swam members are ordered to follow a single target, even temporarily. It was found that *n*th order effects on swarming coherence are least adversely impacted by engineering efforts if the parameters of the canonical boid relationships are altered, rather than introducing new relationships to the system. In effect, preserving simplicity at the individual level, the system exhibits more stable complex behavior at the aggregate level. Future experimentation will examine the effects of increasing levels of autonomy and the implementation of decentralization of higher order functions to ascend through the levels shown in Table 10.1.

It is worth noting that during modeling, it was decided to allow ambiguity and not concern the initial effort with the details of physical propulsion, for example. Fine-grained attributes will be introduced once further paths for uncovering value in application are discovered.

Local data were used to make local decisions (*decide locally*) in a model-free context; the swarm agents never receive control signals from other agents; they merely make decisions based on the activity in their immediate vicinity.

All of this was made possible with *dynamic modeling*, which demonstrates that the emergent swarm itself is resilient to perturbation, even though individual swarm agents are not (*understand emergent properties*).

Unanticipated emergent effects are apparent in the initial attempts to influence and guide aggregate functionality. A better understanding of emergent properties is needed to approach this. Currently, the best methods are trial and error, but future work will address the need to create a repeatable process around this activity. This experimentation created a follower mesh network, requiring rule changes, rather than additions or removal to achieve stability of function.

Future work will involve the development of a more rigorous definition of swarms – one that incorporates more than variance/heading correlations. This will provide more rigorous metrics to evolve toward higher levels of autonomy. We anticipate being able to use this technology for various effects and plan to develop novel rulesets for communications jamming (e.g., cluttering up a radar tower) as well as protection (e.g., protect fighter from surface-to-air missile array).

DISCLAIMER

The authors' affiliation with The MITRE Corporation is provided for identification purposes only and is not intended to convey or imply MITRE's concurrence with, or support for, the positions, opinions, or viewpoints expressed by the authors. Approved for public release: Case: 17-2641.

REFERENCES

Anderson, P.W. (1972) More is different. *Science*, **177** (4047), 393–396.

Bak, P., Tang, C., and Wiesenfeld, K. (1987) Self-organized criticality: an explanation of the 1/f noise. *Physical Review Letters*, **59** (4), 381.

Barry, P.S., Koehler, M.T. and Tivnan, B.F. (2009) Agent-directed simulation for systems engineering. Proceedings of the 2009 Spring Simulation Multiconference. Society for Computer Simulation International, p. 15.

Bar-Yam, Y. (1997) *Dynamics of Complex Systems*, vol. **213**, Addison-Wesley, Reading, MA.

Bar-Yam, Y. (2002) General features of complex systems, in *Encyclopedia of Life Support Systems (EOLSS)*, UNESCO, EOLSS Publishers, Oxford, UK.

Bar-Yam, Y. (2004a) *Making Things Work*, Knowledge Industry.

Bar-Yam, Y. (2004b) A mathematical theory of strong emergence using multiscale variety. *Complexity*, **9** (6), 15–24.

Bar-Yam, Y. (2004c) Multiscale variety in complex systems. *Complexity*, **9** (4), 37–45.

Blanchard, B.S., Fabrycky, W.J., and Fabrycky, W.J. (1990) *Systems Engineering and Analysis*, vol. **4**, Prentice Hall, Englewood Cliffs, NJ.

Bovard, J. (1995) *Lost Rights: The Destruction of American liberty*, Palgrave Macmillan.

Brito, J. and Castillo, A. (2013) *Bitcoin: A Primer for Policymakers*, Mercatus Center at George Mason University.

Carlson, J.M. and Doyle, J. (1999) Highly optimized tolerance: a mechanism for power laws in designed systems. *Physical Review E*, **60** (2), 1412.

Casti, J.L. and Ford, J. (1994) Complexification. *Nature*, **371** (6495), 296.

Durrant-Whyte, H.F., Rao, B.Y.S. and Hu, H. (1990, May) Toward a fully decentralized architecture for multi-sensor data fusion. Robotics and Automation, 1990. Proceedings, 1990 IEEE International Conference on, IEEE, pp. 1331–1336.

Gao, J., Barzel, B., and Barabási, A.L. (2016) Universal resilience patterns in complex networks. *Nature*, **530** (7590), 307–312.

Holland, J.H. (1992) Complex adaptive systems. *Daedalus*, **121** (1), 17–30.

Holland, J.H. (1995) *Hidden Order: How Adaptation Builds Complexity*, Basic Books.

Holland, J.H. (2006) Studying complex adaptive systems. *Journal of Systems Science and Complexity*, **19** (1), 1–8.

Hölldobler, B. and Wilson, E.O. (1983) Queen control in colonies of weaver ants (Hymenoptera: Formicidae). *Annals of the Entomological Society of America*, **76** (2), 235–238.

Jirsa, V.K., Stacey, W.C., Quilichini, P.P. *et al.* (2014) On the nature of seizure dynamics. *Brain*, **137** (8), 2210–2230.

Kauffman, S.A. (2010) *Reinventing the Sacred: A New View of Science, Reason and Religion*, Basic Books, New York.

Lorenz, E. (2000) The butterfly effect. *World Scientific Series on Nonlinear Science Series A*, **39**, 91–94.

Maier, M.W. (1996, July) Architecting principles for systems-of-systems. INCOSE International Symposium, vol. 6, no. 1, pp. 565–573.

Makinson, J.C. and Beekman, M. (2014) Moving without a purpose: an experimental study of swarm guidance in the Western honey bee (Apis mellifera Linnaeus). *Journal of Experimental Biology*, **217**, 2020–202710.1242/jeb.103283.

Miller, J. (2016) *A Crude Look at the Whole: The Science of Complex Systems in Business, Life, and Society*, Basic Books.

Miller, J.H. and Page, S.E. (2009) *Complex Adaptive Systems: An Introduction to Computational Models of Social Life*, Princeton University Press.

Norman, M.D. (2015) Complex systems engineering in a federal IT environment: lessons learned from traditional enterprise-scale system design and change. Systems Conference (SysCon), 2015 9th Annual IEEE International, IEEE, pp. 33–36.

Norman, M.D. and Koehler, M.T.K. (2017) Cyber defense as a complex adaptive system: a model-based approach to strategic policy design. In the Conference of the Computational Social Science Society of the Americas (CSS 2017).

Norman, D.O. and Kuras, M.L. (2006) Engineering complex systems, in *Complex Engineered Systems* (eds D. Braha, A. Minai, and Y. Bar-Yam), Springer, Berlin Heidelberg, pp. 206–245.

Pitsko, R. (2014) Principles for Architecting Inherently Adaptable Complex Systems. Ph.D. Dissertation at the Stevens Institute of technology, Hoboken, NJ.

Rebovich, G. Jr. and White, B.E. (eds) (2010) *Enterprise Systems Engineering: Advances in the Theory and Practice*, CRC Press.

Reynolds, C.W. (1987) Flocks, herds and schools: a distributed behavioral model. *ACM SIGGRAPH Computer Graphics*, **21** (4), 25–34.

Ryan, A.J. (2007) Emergence is coupled to scope, not level. *Complexity*, **13** (2), 67–77.

Ryle, G. (1949) *The Concept of Mind*, Barnes & Noble, New York.

Schollmeier, R. (2001, August) A definition of peer-to-peer networking for the classification of peer-to-peer architectures and applications. Peer-to-Peer Computing, 2001. Proceedings. First International Conference on, IEEE, pp. 101–102.

Sheard, S.A. and Mostashari, A. (2009) Principles of complex systems for systems engineering. *Systems Engineering*, **12** (4), 295–311.

Strogatz, S. (2004) *Sync: The Emerging Science of Spontaneous Order*, Penguin, UK.

Taleb, N.N. (2012) *Antifragile: Things That Gain from Disorder*, vol. **3**, Random House.

Tolk, A., Diallo, S.Y., Padilla, J.J., and Herencia-Zapana, H. (2013) Reference modelling in support of M&S—foundations and applications. *Journal of Simulation*, **7** (2), 69–82.

Tononi, G., Sporns, O., and Edelman, G.M. (1994) A measure for brain complexity: relating functional segregation and integration in the nervous system. *Proceedings of the National Academy of Sciences*, **91** (11), 5033–5037.

Tumer, K. and Wolpert, D. (2000, July). Collective intelligence and Braess' paradox. AAAI/IAAI, pp. 104–109).

Tumer, K. and Wolpert, D.H. (2004) *Collectives and the Design of Complex Systems*, Springer Science & Business Media.

Von Bertalanffy, L. (1972) The history and status of general systems theory. *Academy of Management Journal*, **15** (4), 407–426.

Wilson, E.O. (2000) *Sociobiology*, Harvard University Press.

Wolf, M.T., Rahmani, A., de la Croix, J.P., Woodward, G., Vander Hook, J., Brown, D., Schaffer, S., Lim, C., Bailey, P., Tepsuporn, S., Pomerantz, M., Nguyen, V., Sorice, C., Sandoval, M. and Pomerantz, M. (2017) CARACaS multi-agent maritime autonomy for unmanned surface vehicles in the Swarm II harbor patrol demonstration. Proceedings of SPIE 10195, Unmanned Systems Technology XIX, , 5 May 2017, pp. 101950O–101950O, doi: 10.1117/12.2262067.

AQ2 Zolli, A. and Healy, A.M. (2013) *Resilience: Why Things Bounce Back*, Simon and Schuster. http://www.dodccrp.org/files/NATO_COBP.pdf