WHAT IS A SWARM? A FRAMEWORK FOR UNDERSTANDING SWARMS AND THEIR APPLICATIONS

A Thesis

Submitted to the Faculty

of

Purdue University

by

Zhong W. Thai

In Partial Fulfillment of the

Requirements for the Degree

of

Master of Science in Aeronautics and Astronautics

August 2020

Purdue University

West Lafayette, Indiana

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF THESIS APPROVAL

Dr. Daniel DeLaurentis, Chair

School of Aeronautics and Astronautics

Dr. Shaoshuai Mou

School of Aeronautics and Astronautics

Dr. Kris Ezra

School of Aeronautics and Astronautics

Approved by:

Dr. Gregory Blaisdell

Head of Gambaro Graduate Program of Aeronautics and Astronautics

"14?! Ridiculous! We need to develop one universal standard that covers everyone's use cases." – Randall Munroe, xkcd

ACKNOWLEDGMENTS

A work of this nature is a tremendous undertaking and it would not have been possible for me to complete without the support of many people. Firstly, I would like to thank Dr. Daniel DeLaurentis for taking me under his wing, teaching me new ways to interpret the problems in the world, and giving me opportunities to tackle some of them. The idea for this started out as a simple question he posed during a project brainstorming session, and his continued interest has led to what I hope is a more than satisfactory answer. I would also like to thank Dr. Kris Ezra for his wit and wisdom as a mentor in both the projects we worked on and in the direction of my research, as well as Dr. Shaoshuai Mou for his flexibility in working with me and providing an expert's perspective on multi-agent systems.

In addition to my advisors and mentors, I would like to thank my friends and colleagues who breathed life into my time at Purdue. Andrew Haines, Chris Brand, and Prajwal Balasubramani were instrumental in helping me coauthor two papers and creating the simulation code used in this work. They have also been valuable sounding boards, helping me refine raw, confusing thoughts into clean, defined ideas. Melanie Grande was also an inspiring colleague and friend that helped motivate me to work harder and an unyielding critic who pointed out flaws in my reasoning. And of course, the many others in the lab who were willing to tolerate my puns and helped me along the way.

Lastly, I would like to thank some of my friends whose counsel and late night conversations kept me sane and helped me adjust to a different place. Their willingness to maintain a friendship despite the turbulent changes in our lives and their support through difficult times is something I will be eternally grateful for.

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

ABBREVIATIONS

ABM	agent-based modeling
BDI	beliefs, desires, and intentions
INCOSE	International Council on Systems Engineering
MAS	multi-agent system
MRS	multi-robot system
OODA	observeorientdecideact
OOP	object-oriented programming
PLEXNet	planetary exploration network swarm architecture
P2P	peer-to-peer
ROPE	resources, operations, policies, and economics
RPO	robust portfolio optimization
SEBoK	Systems Engineering Body of Knowledge
SODA	systems operational dependency analysis
SoS	system-of-systems

GLOSSARY

agent	individuals that can function independently and cooperate with other agents
characteristics	values and parameters defining an agent, swarm, object, or environment
disruption	events that change the condition or state of agents,
environment	the space in which agents exist and operate objects, or environments
objects	non-agent entities that agents are able to interact with and modify
objectives	performance metrics that influence how tasks should be accomplished
purpose	the overall reason a swarm exists
swarm	a group of interactive agents working together to accomplish specific goals and purposes
task	a specific action that an agent or swarm is to perform

ABSTRACT

Thai, Zhong W. M.S., Purdue University, August 2020. What is a Swarm? A Framework for Understanding Swarms and their Applications. Major Professor: Daniel A. DeLaurentis.

As problems in the world become increasingly complex, designers in multiple disciplines have begun to propose swarms as a solution. The espoused benefits include flexibility, resilience, and potential for decentralized control, yet there lacks consensus on what a swarm is, what characteristics they possess, and what applications they are able to address. This study addresses these questions by creating a unified approach for understanding and analyzing swarms, called the Swarm Analysis Framework. The framework pursues three goals: 1) provide extensive analysis on the many characteristics and applications that define a swarm, 2) remain flexible enough to facilitate design, testing, analysis, and other problems in understanding swarms, and 3) outline swarm applications specific to aircraft and spacecraft based swarms. Afterwards, the Swarm Analysis Framework is used to guide a case study in which the application is a swarm was developed to study one of these aerospace applications. Ultimately, the Swarm Analysis Framework, along with its extensions improvements, should be able to act as a guide or roadmap in understanding how swarms behave across multiple disciplines.

1. INTRODUCTION

One of the most commonly proposed solutions to complex problems in recent years is the swarm. These are often described as flexible, scalable, resilient, and most importantly, cheap, amongst other things. Despite the many claims of being a panacea to the problems of an increasingly complex world, there is little consensus on what even constitutes a swarm, let alone what different kinds exist and what problems they are actually capable of solving. This is not to say that there is no significant body of work study swarms. In fact, the opposite is true, with extensive studies on different swarm aspects. What is commonly missing however, is a way to bridge these many fields and craft a unified response to a simple question: What is a Swarm?

Before delving into that question, it is important to understand why this question even matters and why swarms are relevant. Many problems today are solved using monolithic systems. These are highly specialized, self-contained systems designed to solve a variety of problems on their own or through the guidance of people. In some cases, they are a perfect solution to the problem they were designed for. Unfortunately, they have two distinct drawbacks: cost, risk, and scale. Because monolithic systems are expected to solve many possibly conflicting problems, they often involve devilish integration problems that require ingenuity, time, and luck to solve, while directly translates to cost. Because there is only one system, more emphasis is placed on system reliability as well. Should the monolithic system fail, then the entire system fails, leading to higher margins of safety and reliability, which translates into even more cost. In addition, due to physics, there are some problems they simple cannot solve due to scale. There is simply an upper limit to the amount of force or cover a single monolithic system can provide no matter what miracles engineers manage to achieve.

The solution to this is to use multiple systems. With such a distributed approach, there is simply less of a need to make a single system capable of addressing every possible requirement, lowering integration costs. Specialized systems may even be able to address certain problems in better ways than a single monolithic system. In addition, with multiple vehicles, some functions can be duplicated, which reduces the chance for a total mission failure if a single system fails. Overall, this redundancy may result in higher costs, but in return, failure goes from a punishing binary to a more forgiving scale. With multiple systems, problems limited by scale can once more be solved. Multiple agents translate to more output that would have been impossible with a single monolithic system. As the number of systems increases, this group of systems eventually starts to become its own entity, sometimes referred as a swarm or multi-agent system.

This conclusion was reached by many people across different fields. Satellites became constellations in order to increase coverage, while computer processors began to come with multiple cores in order to increase computation time. Over time, many fields developed their own unique ways to use multiple systems to increase performance. As the world become more interconnected however, some of these established domain specific swarm methods were starting to be insufficient at solving increasingly complex problems. Fortunately, solutions were probably developed to solve similar problems. The issue is that these solutions were probably developed in a completely different field.

As technology becomes cheaper and swarms and other multi-agent systems become easier to develop, this problem will only exacerbate. At first glance, the solution seems obvious: explore different fields until a solution is found. In reality this is difficult to do. To identify these methods and solutions, a designer would need to gain requisite knowledge in a separate field just to even know where to look, assuming the answer is in that field to begin with. This is further complicated by the lack of agreement in academia on the definition of a swarm.

At this point, the problem circles back to the initial question of defining a swarm. To answer this question, a unified framework for understanding the general concept of a swarm will be developed using the principles across the many disciplines that have already studied swarms.

1.1 Goals

The primary goal of this study is to create a framework that can be used to understand and analyze swarms from many disciplines and perspectives. To do this, two other questions must be answered: "What are the characteristics of a swarm?" and "What can a swarm do?" By doing answering these questions, a comprehensive approach can be developed to systematically understand the specific parameters that differentiate different swarms from each other and the specific tasks these swarms are best suited for. An additional benefit of this goal is to compile a body of knowledge that can be used to guide new designers in understanding a swarm and its applications.

In addition to this goal, this framework needs to be expansive enough that it can be used to solve a variety of swarm problems from understanding the mechanisms of existing swarms to developing new swarms to solve specific problems. To aid with swarm design, it cannot be structured around any specific design methodology without restricting its potential use outside the limitations of that particular approach. Despite this use, the framework itself is not solely meant for designing and crafting swarms for specific problems. The goal is to create a structured guide that can assist in swarm design, as well as help identify features of existing swarms, guide studies and analyses of certain swarm features on performance, and other problems that involve swarms. Fundamentally, it is a list of swarm features and applications, not a method.

Lastly, even though the framework should be general enough for use across disciplines, this study should be able to use this the framework to analyze swarm problems in aerospace. With the advent of low-cost aircraft and spacecraft like commercial quadcopters and cubesats, there is a pressing need to study the potential applications of swarms to better utilize these new platforms to their fullest potential.

In the remainder of this chapter, a lexicon will be developed to craft a loose definition for a swarm that can be used across multiple disciplines, alongside other words that will be used extensively in this study. The next chapter will describe some of the research that has been done towards swarms over the years and evaluate existing approaches that describe a swarm's characteristics and applications. The next two chapters will describe a unified approach created to address these research goals, called the Swarm Analysis Framework. This framework will be split into two parts: one that describes the different characteristics of a swarm in Chapter 3, and one that describes swarm applications in Chapter 4. Once the Swarm Analysis Framework is described, Chapter 5 will illustrate how it can be used to analyze a specific swarm problem and observe swarm behavior. Useful references and supplementary information can be found in the appendices. Due to sheer scope of the main question, the idea of a swarm will not be fully defined, but a great many of its defining ideas will be established by answering its follow-up questions. By the end of this study, the three principle questions and three goals outlined in this chapter should be addressed in extensive detail.

1.2 Definitions

To understand swarms across domains in this study, it will be useful to develop a lexicon to reduce confusion. The most important definition within this lexicon will be the definition of a *swarm*. As much of the early swarm research is in biology, so too were the earliest definitions. These definitions are simple, like the Oxford definition:

> **Swarm** – *"A large or dense group of insects, especially flying ones."* [1]

Merriam-Webster expands on this a bit by considering inanimate objects as well:

> **Swarm** – *"A large number of animate or inanimate things massed together and usually in motion."* [2]

Aside from these, swarm can also refer to a group of honeybees in the process of starting a new colony, but the definitions do not go much farther.

As mobile robotics developed and concepts for group of robots began to take shape, definitions migrated along with key findings used to facilitate these concepts. From there a diaspora of definitions took root, but eventually the concept, at least in robotics, began to settle. Some definitions for swarms in the multi-robot systems (MRS) or multi-agent system (MAS) sense are listed below:

Swarm – *"A large number of autonomous robots utilized to realize a distributed system."* – Iocchi 2001 [3]

Swarm – ***a:*** *"A bunch of small cheap dumb things to do the same job as an expensive smart thing."* ***b:*** *"A collection of autonomous individuals relying on local sensing and reactive behaviors interacting such that a global behavior emerges from the interactions."* – Clough 2002 [4]

Swarm – *"Large number of relatively simple robots."* – Bayindir 2007 [5]

Swarm – ***a:*** *"A group of agents that collectively accomplish tasks normally requiring intelligence without central control."* ***b:*** *"Any swarm capable of swarm intelligence."* – Beni 2019 [6]

Multi-agent system – *a decentralized, loosely coupled network of problem solving entities that work together to find answers to problems that are beyond the individual capabilities or knowledge of each entity* – Flores-Mendez 1999 [7]

Of these definitions, the only consistently mentioned trait is *large numbers*. Other traits that appear include autonomy, cooperation, distributed/decentralized systems, low cost, or low intelligence, but these traits are not universally agreed upon. This is also reflective in other papers where the definition of swarm or MRS is implicit.

Parallel to development of swarm robotics, the concepts of swarms took a different meaning in systems involving people. For warfare, swarming began to refer to a coordinated group capable of "sustained pulsing" from all directions, as elaborated in Arquilla and Ronfeldt's *Swarming and the Future of Warfare*. According to them, two requirements need to be met for swarming: large numbers of units that communicate and coordinate to perform maneuvers, and the ability to act as sensors for the entire swarm [8]. Though primarily focused on warfare, Arquilla and Ronfeldt emphasizes that this idea of swarming is not exclusive to military and even applies to social movements.

In order to capture a general definition for swarms from multiple domains, the swarms discussed in this study will be primarily defined as:

Swarm – *a group of distinct agents that work together towards some common purpose or set of purposes*

Note that the definition as used in this study is open ended and not meant to be definitive. Because the idea of swarms often overlaps with ideas of multi-agent systems, groups of agents, or systems-of-systems, it would be more useful to keep the definition flexible enough to be used with these other areas wherever relevant.
In addition, several other definitions will be used in this study to ensure clarity.

Agent – *a self-contained entity that acts upon its environment possessing the ability to act independently of others*

Objects – *non-agent entities that agents are able to interact with and modify*

Environment – *the space in which agents exist and operate*

Disruption – *events that change the condition or state of agents, objects, or environments*

Function – *things agents are capable of doing*

Characteristics – *values and parameters defining an agent, swarm, object, or environment*

Task – *a specific job that an agent or swarm is meant to do*

Operation – *methods used by an agent or swarm to accomplish a task*

Purpose – *the overall reason an agent or the swarm exists*

2. LITERATURE REVIEW

Due to the expansive nature and extensive application of swarms, there is active research across many fields including biology, robotics, and optimization. Many of these studies tend to focus on a particular area and have varying motivations and goals. As a result, in order to craft a framework that captures the general characteristics and applications of swarms, many different areas are explored to get a comprehensive picture on what defines a swarm and what it can be used for. For this, various studies of proposed swarm taxonomies are examined to evaluate their effectiveness in capturing the complexities of a swarm. Studies from biology, the military, computer networks, robotics, and system-of-systems (SoS) are also examined to gain context of what a taxonomy would need to encompass. Because this study is focused on a high-level understanding of swarms, studies that sought to understand swarms behavior analytically are not explored in detail.

2.1 Taxonomic Research

One of the earliest attempts at classifying different swarm architectures was done in 1993 by Dudek et al. [9], who divided swarms into seven axes, shown below in Table 2.1.

Each axis includes cases covering possible configurations within each axis. *Collective Size* ranges from single robots up to effectively infinite robots. *Communication Range* spans from none, to local, to infinite communication. *Communication Topology* explores ways to structure communication paths, including unrestricted methods like broadcasting or direct agent-to-agent communication and restricted methods where information goes through a hierarchy or other network structure. *Communication Bandwidth* describes the cost of transmission from no cost to prohibitively expensive,

Table 2.1.. Dudek et al. 1993 Taxonomic Axes [10]

Axis	**Description**
Collective Size	The number of robots in the environment.
Communication Range	The max communication distance between robots.
Communication Topology	Which robots in range can be communicated with.
Communication Bandwidth	How much information robots can transmit.
Collective Reconfigurability	The rate at which the swarm changes.
Processing Ability	The computational model utilized by robots.
Collective Composition	Swarm robots homo- or heterogeneity.

with a special case where communication is done through physical motion. *Swarm Reconfigurability* describes not only the speed at which the swarm topology changes, but if these changes are coordinated. *Swarm Processing* focuses on the computation model used, ranging from simple non-linear summation up through Turing machines. Lastly, *Swarm Composition* describes the homogeneity of the entire swarm. The rest of the paper [9] and Dudek's later 1996 paper [10] then uses this taxonomy to identify combinations that work well for certain tasks like modeling a Turing Machine, mapping a graph, and self-location and organization.

Several year later, Cao et al. [11] sought to provide a taxonomic organization of literature up until that point, expanding from swarm architectures to problems and solutions which can be seen in Table 2.2. The first four axes focus on the cooperative behavior while the last axis focuses on applications of this cooperative behavior.

Table 2.2.. Cao et al. 1997 Research Axes [11]

Axis	**Description**
Group Architecture	The infrastructure underlying a group of robots.
Resource Conflicts	How robots resolve conflicts.
Origins of Cooperation	How cooperation is motivated and achieved.
Learning	Adaptability and flexibility of the group of robots.
Geometric Problems	Research issues from using robots in a physical space.

These axes are further broken up into several categories. *Group Architecture* includes key defining architectural features, which are detailed in Table 2.3. Resource conflicts encompasses cases where robots share space, manipulable objects, or communication media. *Origin of Cooperation* is divided into either eusocial behavior, where cooperation is an indirect result of individuals acting to survive, or cooperative behavior, where cooperation is deliberate. *Learning* iss not elaborated in detail as rigorous techniques did not develop until recently. *Geometric Problems* details studies on formation and marching problems.

In contrast to Dudek and Cao, Iocchi et al. [3] developed a taxonomy with a top-down approach, grouping classifications in a hierarchy, as shown in Figure 2.1. In Iocchi's taxonomy, multi-robot system structure is first divided into *Cooper- tive* and *Uncooperative* systems by whether or not agents in a swarm cooperate to accomplish their tasks and goals. *Cooperative* systems are further divided into *Aware* and *Unaware* systems based on whether individual robots have knowledge of other agents in the system. *Aware* robots are broken down by *Coordination*, where strong coordination corresponds to systems dependent upon a coordination protocol, weak

Table 2.3.. Cao et al. 1997 Key Features of Group Architecture Axis [11]

Axis	**Description**
Centralization	The control structure of the architecture, including hierarchy of robots.
Differentiation	Homogeneity of the group of robots. Heterogeneity introduces complexity.
Communication Structures	Modes of inter-agent interaction. Via environment, sensing, or communication.
Modeling of Other Agents	Ability to model to beliefs, desires, intentions, and states of other agents.

coordination corresponds to systems that can coordinate without a protocol, and no coordination. *Strong Coordination* is then further broken down in terms of *Centralization*, where centralized systems involve "leader" robots and distributed systems feature autonomous robots with no leader. In addition to this structural hierarchy, two additional axes are described: *Communication*, divided into direct and indirect communication, and *Homogeneity*, describing where or not agents are identical.

Iocchi et al. also includes two definitions to describe a robotic swarm's response to the dynamics of their environment, *Social Deliberation* and *Reactivity*, and assigns these to the endpoints in the structural hierarchy. *Social Deliberation* would correspond to a coordinated response where the team reorganizes members using some strategy to best take advantage of available resources to accomplish a task. *Reactivity* on the other hand is more individual and responds to changes by having individuals adjust and reorganize instead of using some larger strategy.

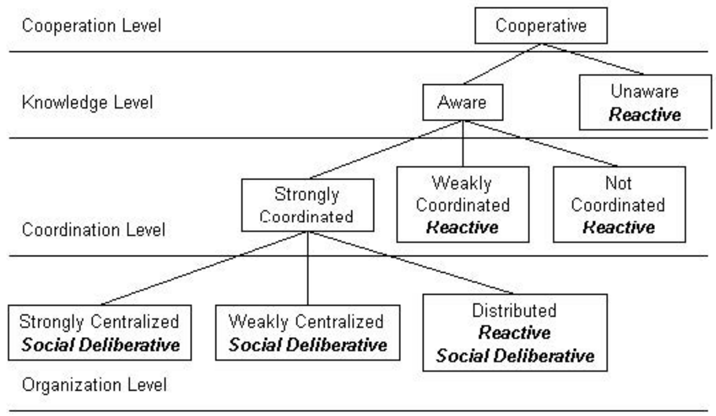

Figure 2.1.. Multi-Robot System Taxonomy [3]

All three of the taxonomies developed by Dudek, Cao, and Iocchi all manage to cover many elements that are useful for describing swarms. However, the way these characteristics are organized has key drawbacks. Dudek's taxonomy oversimplifies the complexities within a swarm, reducing characteristics into rigid and separate axes. Such a method equates simple measurable parameters like *Collaective Size* and *Collective Composition* to the much more complex idea of *Processing Ability*, effectively overlooking any of the nuances of the entire field of swarm intelligence. These axes also have an implication of orthogonality, which is not necessarily true. Many of the axes have components that are frequently coupled. For instance, *Communication Topology* often places limits on what options are available for *Processing Ability* and directly impacts latency of the swarm within *Collective Reconfigurability*. Though the taxonomy captures many parameters of communication, the overall taxonomy's small scope lacks the coverage that Cao's taxonomy exhibits.

In contrast to Dudek, Cao's taxonomy is less rigid and broader. Instead of strict axes, Cao grouped similar parameters into categories, which can be broken down further into subcategories, which allow for these groups to be expanded and refined. This also introduces new concepts like centralization and breaks down processing ability into ideas like internal modeling and coordination between agents. However, the taxonomy also does not address axes coupling in any way.

Iocchi's taxonomy begins to address links between parameters using a hierarchical breakdown. By breaking down a highly general characteristics of swarms like *Cooperation* into spectrums, and then breaking these down further, Iocchi acknowledges some of the common trends seen with swarms and alludes to the emergent nature of swarms. However, this hierarchy is conceptual and has a limited and rigid scope that only focuses on cooperative robotic swarms.

These taxonomies also suffer from several other issues. The criteria for establishing these axes are not clear, making expansions to them to encompass a general swarm difficult. These taxonomies also focus almost exclusively on the swarm level, and as a result tends to feature aggregate characteristics like size or centralization. This misses the impact of agent specific characteristics like individual capabilities and the types of relationships agents share with other agents. Because a swarm is an emergent entity that can only exist as a result of its individuals, a taxonomy needs to be able to include the ways individual agents are different in order to characterize generic aggregate properties like homogeneity, centralization, and size.

2.2 Swarms in Biology

Given that much of swarm research has roots in biomimicry, studies in biology are examined to gain possible insights on how swarms function in nature and to give context outside of studies in robotics. Some of the earliest studies that mention swarms are focused on beekeeping in the late 19th century and mold spores in the early 20th century, but research into the mechanisms that drive swarm behavior did not begin until the late 1950s when communication in bees was found to be through the use of "bee dances" signaling distance and direction for a target [12].

A later study from Goss (1989) showed how ants can find optimal paths over time when foraging without any prior knowledge of coordination [13]. When individual foraging ants search for food, they leave a pheromone trail, marking paths to food sources for later use. Individually, there is no way to identify any optimality. Over time and across all of the foragers however, shorter paths end up being traversed more often by an ant than longer paths, creating a stronger pheromone trail over time, attracting more ants to that path and creating a positive feedback loop. Eventually an optimal path is found. This concept was eventually used to develop a metaheuristic optimization technique in 1992, aptly named ant colony optimization [14].

A separate study from Gordon (1996) examined how ant colonies handled tasking [15]. Essentially, ants release different pheromones when performing a specific task like foraging for food, searching for building materials, or constructing parts of the nest. Across an entire colony this results in a specific pheromone mixture, for other ants to interpret. If a particular pheromone is missing or in abundance, then ants in the imbalanced region can choose to change roles to correct for this imbalance. This effectively allows a colony to control tasking without the need to count how many ants are performing a task and introduces the idea of pheromones which can potentially be used by robots as a mechanism to facilitate complex tasks without the need of an coordinating authority.

Outside of insects, studies of swarming behavior have also been done on birds, specifically on flocking. In 1987, Craig Reynolds introduced the artificial life concept of boids [16]. In the simulation, bird-oids, or boids, would fly around the simulation environment under a set of rules: separation, alignment, and cohesion. By dictating the distance boids would maintain between each other, the variance of their direction vectors, and the direction of each boid towards the centroid of all boids, Reynolds was able to duplicate flocking behavior. This model is also able to add additional rules, allowing for complex behaviors to be modeled and studied.

Overall, most of the studies in naturally occurring swarms revolved around identifying mechanisms that drove behaviors. Both ant studies are based on the use of pheromones to facilitate decentralized tasks within a swarm, and boids introduced rulesets that would define a rudimentary set of beliefs, desires, and intentions (BDI) for an agent. However, aside from showing mechanisms that can apply to other types of swarms, these studies had a limited scope and did not explore the general concept of what defines a swarm. Nonetheless, they provide unique context and give examples of swarms that would need to be encompassed by a general swarm design framework.

2.3 Applications of Swarms

Much of the interest in swarms is driven by their potential applications, so it would be vital to examine these applications to gain additional context with respect to their end goals. Research in this area is vast, with new applications discovered or created by problem solvers every day. Fortunately, a study published in 2016 by Bayindir et. al. presents an extensive taxonomy that organizes over 150 different studies of robotic swarms grouped into specific tasks, listed in Table 2.4. Each grouping included design methods, relevant past work, and mathematical models and metrics specific to the task [17]. In addition to these main tasks, several others are described in high level terms, shown in Table 2.5. Collectively, these tasks covered within these groups span much of the design space for robotic swarms.

Some of these task categories overlap with previous studies by Mohan and Ponnambalam 2009 [18] and Navarro and Matia 2013 [19]. Across these studies, swarm tasks are consistently being grouped into similar categories based on specific research problems: agent positioning, manipulation and transport, search, tasking, or control.

Whereas Bayindir, Mohan and Ponnambalam, and Navarro and Matia divided swarm tasks by problems researchers have attempted to solve, Balch (1999) attempts to decompose these multi-robot tasks into separate axes in a similar manner to Dudek's (1993) taxonomy of swarms. Balch proposes two taxonomies, one that decomposes applications by different task components, and one describing reinforcement functions or reward functions for the swarm. These can be seen in tables 2.6 and 2.7.

Table 2.4.. Bayindir et al. 2016 Taxonomy of Swarm Robotics Tasks [17]

Application	**Description**
Aggregation	Gathering a number of autonomous individuals in a common place
Flocking	Large groups of individuals moving together toward a common target location
Foraging	Finding items scattered in the environment and bringing them to specific locations
Object Clustering and Sorting	Grouping scattered items together into separate sorted groups
Navigation	How a *single* limited sensing robot reaches a target in an unknown location with the help of other robots
Path Formation	Collectively building a path between two locations in an environment to limit travel time
Deployment	Individual self-tasking in an environment without central coordination
Collaborative Manipulation	Tasks where robots must work together to manipulate an object in the environment
Task Allocation	Dynamically changing tasks executed by each robot based on local perception of the environment
Other Tasks	Studies surveyed for this taxonomy that did not fit into the above categories

Table 2.5.. Bayindir et al. 2016 Additional Swarm Robotics Tasks [17]

Application	Description
Odor source localization	Finding the source of an odor in the environment by tracking odor and wind direction
Object Assembly	Building structures from objects located in the environment
Self-assembly and morphogenesis	Autonomously connect to each other using local interactions and potentially create shapes
Coordinated Motion	Moving a modular robotic structure by having robots move in a common direction
Group Size Estimation	Obtaining group size in a distributed manner without direct communication
Distributed Rendezvous	Agreeing on a location to converge while maintaining a connected graph
Collective Decision Making	Converging to unanimous decisions from multiple options with different rewards
Human-swarm interaction	Ways for humans to impact the swarm without directly controlling every robot

In addition to the tasks explored by Bayindir, Mohan and Ponnambalam, Navarro and Matia, and Balch, which tend to be cooperative and in neutral unchanging environments, additional tasks exist in a military context. In a 2000 book from the RAND Corporation, Arquilla and Ronfeldt discuss the history of military doctrine and the use of swarming in the future. Swarm in this case refers to a newly emerging,

Table 2.6.. Balch 1999 Taxonomy for Robot Tasks [20]

Descriptor	**Meaning**
Time	time to complete a task
Criteria	how performance is measured (average vs total)
Subject of action	whether focus is on swarm agents or objects
Resource limits	scarcity of resources and if agents compete
Group Movement	how agents move for task
Platform capabilities	minimum capability and number of agents required

Table 2.7.. Balch 1999 Taxonomy for Robot Reinforcement Functions [20]

Descriptor	**Meaning**
Source of reward	if reward is from sensors or determined by other agents
Relation to performance	if rewards are from performance of heuristics
Time	time delay for reward
Continuity	if reward is discrete or continuous
Locality	if rewards are localized or for all swarm agents

specific form of engagement where autonomous or semiautonomous agents converge on and assault a target in a highly coordinated sustained pulsing pattern meant to seem like an attack is chaotically coming from all directions [8]. Though restricted in scope, this is a nonetheless a novel swarm task.

Other military swarm tasks are explored in separate studies by Clough (2002) and Scharre (2015). Clough proposes that swarms would be best equipped to handle tasks like area search and attack, surveillance and suppression, psychological warfare, diversion, software reduction, and survivability [4]. Scharre provides suitable swarm tasks like coordinated attack and defense, dynamic self-healing networks, distributed sensing and attack, deception, and swarm intelligence [21].

Whereas the robotic tasks are primarily focused on a single swarm, military applications consider adversaries and competing entities. The idea of area coverage or deception both come from ideas like having enough swarms to completely overwhelm the sensing capacity of a target, which is a similar concept to collaborative manipulation, except with information instead of physical objects. Meanwhile, psychological warfare, diversion, and coordinated defense are concepts that only exist when there is an external concept or entity that must be considered like morale, secrecy, or hostile adversaries.

Overall, these studies provide an overview into the vast problem space that swarms are used to address. Bayindir provides an extensive list of common tasks with vastly different requirements and considerations, alongside appropriate methods and metrics and past studies. Arquilla and Ronfeldt, Clough, and Scharre provide additional applications, broadening the problem space for more generalized swarms. In order to account for the new contexts however, Bayindir's taxonomic method would be insufficient, and this would amount to adding more task groups, eventually making the taxonomy into an unwieldy organization tool as new tasks develop. A new organization tool that is simultaneously able to capture these many contexts but remain simple and compact would need to be developed if a framework for general swarm design is to be created.

2.4 A System-of-Systems Approach

Because many swarms feature independent and autonomous agents, they can be considered systems-of-systems (SoS) using Maier's principles of operational and managerial independence [22]. Swarms often exhibit and feature other SoS traits like geographic distribution, evolution, and emergent behavior described by DeLaurentis (2005) [23]. As a result, SoS design principles and organizational tools can also be used to address swarms. One such tool is the ROPE table introduced by DeLaurentis (2005).

The ROPE table is an organizational framework that organizes a SoS problem along two axes: type of system and scope. SoS problems often consist of many types of systems and entities, which largely fall into four categories with similar roles in an SoS: resources, operations, policies, and economics. Scope breaks these systems down further into hierarchical levels, designed by greek letters (α, β, γ, etc.), where α-level systems consist of the base level entities under consideration, β-level systems consist of collections of α-level systems, and progressively high levels consist of collections of the systems preceding it.

In the larger context of solving problems, a swarm would fall in the α- and β-levels of an actual SoS ROPE table. Despite this, swarms can be considered using similar principles. Using a scope-based hierarchical breakdown, characteristics of a swarm can be broken down into two levels, encompassing both low-level agent parameters as well as high-level swarm parameters while also maintaining the connections between the two. A middle level between this low and high level can also be used to address relationships between agents, and additional higher levels can capture increasingly complex characteristics like interactions between swarms. Such a breakdown is highly useful; complex and nuanced characteristics like swarm intelligence can now expand into appropriate levels. The categorical breakdown, however, would not be as useful when defining the key swarm characteristics at the scope of this study, and would be better to address the overall problem the swarm is seeking to solve.

Swarms can also be considered using network theory, where swarm a swarm can be described as a graph where agents correspond to nodes and relationships between agents correspond to links [24]. With this modeling approach, many characteristics of a swarm can be studied just from looking at how agents are arranged using a variety of metrics [25, 26]. This is discussed in greater detail in Chapter 3.4.3.

—

Overall, several conclusions can be drawn from existing literature:

1. Taxonomies used to describe swarm characteristics and differentiate swarm types exist but suffer from limited scope, rigid organization, missing context.

2. Studies in biology are focused on mechanisms of swarm behavior in animals and do not provide much useful work with respect to defining features of swarm architectures but does provide context that must be included in a general understanding.

3. There is extensive work in identifying methods and metrics for specific swarm tasks and applications in both robotics and military domains, but the best taxonomy is not organized in a manner that scales to multiple domains.

4. A system-of-systems approach, specifically through concepts like organization by scope and approaches like agent-based modeling, may be able to create a framework that is able to address these limits in current research and give a better understanding of how a swarm works.

3. SWARM ANALYSIS FRAMEWORK – SWARM CHARACTERISTICS

In order to provide a systematic way to understand the characteristics and applications of a swarm, this chapter and the next outlines the Swarm Analysis Framework. This framework consists of two taxonomies, one that analyzes and organizes the many characteristics and parameters that define and influence a swarm and its behavior (Taxonomy of Swarm Characteristics) and one addressing the different applications that swarms can address (Taxonomy of Swarm Applications). Collectively, this provides a way to analyze a swarm or its applications in extensive detail and can be used in conjunction with swarm design methodologies to develop agents, swarm architectures, or methods to solve problems. As of now, the scope of the framework is limited to a single swarm and is unable to fully describe interactions between multiple swarms. The framework also does not fully address characteristics and behaviors of the environment swarms operate in. These can be developed in future work, however.

The rest of this chapter describes the Taxonomy of Swarm Characteristics in extensive detail from the logic behind how it is organized to examples of each of the concepts described. The next chapter does the same for the Taxonomy of Swarm Applications, and Chapter 5 applies the entire Swarm Analysis Framework to a case study to illustrate how it can be used to solve a single swarm problem.

3.1 Organizing Swarm Characteristics by Scope

Because of the expansive design space for swarms, it is important to have a tool to understand the primary characteristics of a swarm and how they interact. Such a tool can then be used to distinguish between different swarm types and establish archetypes with common methods, tasks, and considerations. Many have attempted to create such a tool, like Dudek [9], Cao [11], or Iocchi [3], but their attempts are limited in scope and only capture a small part of a much larger picture. The Taxonomy of Swarm Characteristics seeks to rectify this by organizing characteristics by scope and purpose.

In contrast to the taxonomies presented by Dudek, Cao, or Iocchi, a new taxonomy for swarms can be created using a principle similar to the levels from DeLaurentis's ROPE Table [23]. This is useful for dividing sprawling systems-of-systems (SOS) into distinct levels for individual study. A similar idea is seen in military doctrine in the tactical, operational, and strategic levels of warfare, which is used to separate low-level, localized, short-term tactical decisions from high-level, holistic, and long-term planning [27–30]. Similar scales exist for swarms, so organizing by scope using levels allows similar analyses.

Swarms can be interpreted along three levels: *Individual*, *Interaction*, and *Group*. The *Individual* level focuses exclusively on aspects of a single swarm agent, like hardware specifications or agent behavior. The *Interaction* level focuses on the relations between a single agent and another peer, like superior-subordinate and peer-to-peer (P2P) relations and methods of communication between agents. The *Group* level focuses on the aspects that only begin to exist with multiple agents, like homogeneity or swarm intelligence. Additional levels exist, but these are beyond the scope of a singular swarm. Any lower level would essentially include agent design and provide too much fidelity, while higher levels would add too much complexity for the scope of the framework.

This breakdown can be likened to networks and graphs. If a single swarm constitutes an entire graph or a component of a larger graph, individual agents correspond to nodes, while their relationships with other agents correspond to links. Characteristics describing these individual agents are within the *Individual* level while characteristics describing these links are considered within the *Interaction* level. This can be visualized in figure 3.1.

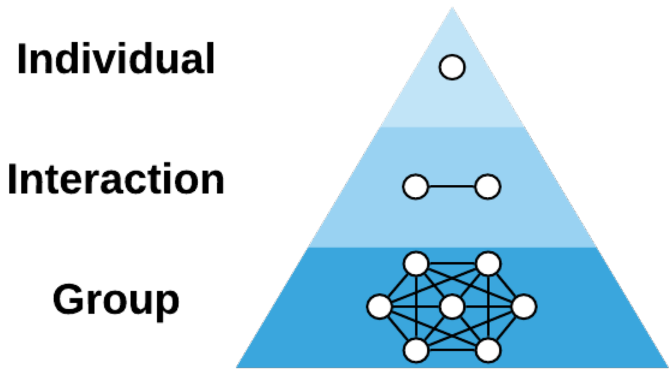

Figure 3.1.. Levels of a Swarm

This breakdown has several benefits. Swarms are a complex, emergent entity that formed by its constituent agents and their interactions, and their characteristics feature this complexity and emergence as well. *Individual* and *Interaction* level characteristics like communication range or linking rules directly allow or strongly impact *Group* level characteristics like networks, multi-agent processing, and dynamic responses. These lower level characteristics provide a way to show how interconnected many parameters are within a swarm and can identify causes to behavior like coupling of characteristics.

Allowing swarm characteristics to span across multiple scope levels also allows highly complex concepts to be studied in detail at various levels, while giving room for nuanced analysis of simpler parameters. Using this taxonomic structure, swarm control can be broken into a study of power dynamics between two peers or a superior and subordinate at the *Interaction* level and a separate study on hierarchical centralization at the *Group* level. Meanwhile, homogeneity can expand to look at common traits among different agents instead of just looking at what percent of a swarm is identical.

By considering both component and aggregate level properties, this approach also allows for top-down, bottom-up, mimicry-based, and evolutionary approaches to understanding swarms [31]. Desired *Group* level characteristics can be analyzed to find requirements for individual agents and behavioral patterns in order to create this behavior. On the other hand, a collection of agents with a set of characteristics can be studied to find favorable interactions to create beneficial emergent swarm behavior. In the process, methods appropriate to a certain level can be tested.

Each level is divided into different categories which contain a set of parameters that address a particular aspect of a swarm. These are described in this chapter alongside examples showing how agents or swarms can be interpreted. These is further divided into additional subcategories covering specific aspects of that characteristic. Though they address different concepts, these categories are often overlap and are linked together in numerous ways due to shared hardware or logic, dependencies, or coupled behaviors. Choices in one domain often impact other others within interconnected systems. As such, categories also include descriptions on how they impact other categories.

The following sections explore each of the three levels and their categories and subcategories, along with how these categories interact. The end of each level also includes relevant examples to illustrate the taxonomy. The taxonomy (excluding subcategories) can be seen in figure 3.2. A reference version including subcategories and summary statements can be found in Appendix A.

The described subcategories for each category are not a fully exhaustive list, but a general set of subcategories common to many agents or swarms. Depending on the complexity and requirements of an agent or swarm, additional subcategories can be added and existing ones can be split to describe agents in more detail, provided they are relevant and have a significant impact on the swarm.

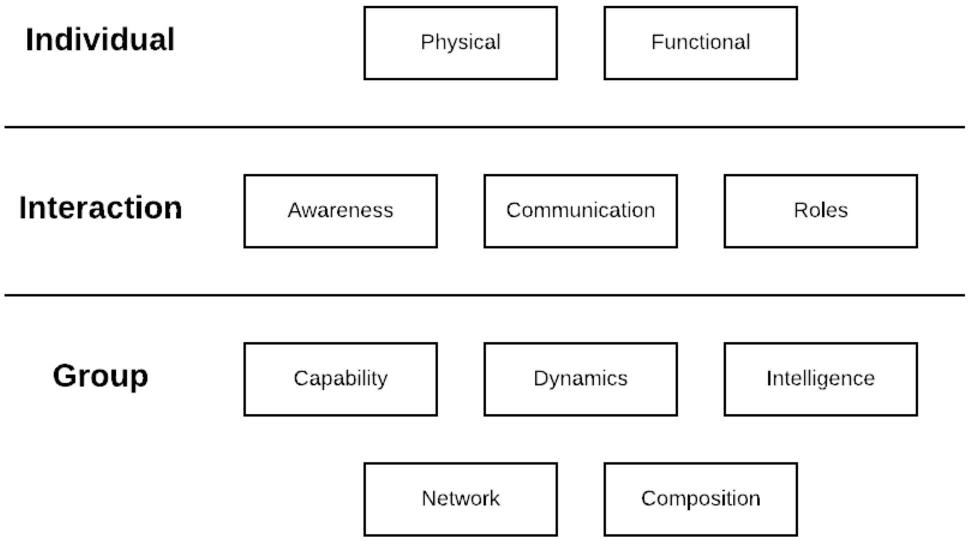

Figure 3.2.. Taxonomy of Swarm Characteristics

A Brief Comment on Resource Demands and Costs

Before describing the categories in the taxonomy, it is important to note that this taxonomy does not address resource demands aside from notional power and maintenance demands. This is because resource demands exist at each scope and do not describe the characteristics of a swarm, only its secondary needs. The Taxonomy of Swarm Characteristics focuses on the *Resource* and *Operations* categories of the ROPE table, while resource demands and costs are aspects within the *Economics* category (*Policies* address external impacts which are not studied in this work). Nonetheless, it is useful to acknowledge the different types of resources that may be required to operate a swarm as this puts an upper bound on a swarm.

Resource demands and costs can largely be divided into five types: money, equipment, people, consumables, and information [32]. Money describes the financial cost of creating, maintaining, and operating the swarm, and is also a secondary feature of the other four types which also cost money. Equipment is the supporting infrastructure needed to manufacture and maintain swarm components, people are the number and types of individuals required to operate the swarm, consumables include fuel and materials needed for manufacture and repair, and information is the basic knowledge required to build or operate the swarm. All of these can strongly impact the purpose and realization of a swarm at all levels and need to be considered at all times. These are also addressed to some extent as metric in Chapter 4.

3.2 Individual

The *Individual* level is focused entirely on the parameters and functions of a single agent, establishing its capabilities and limitations. Much like nodes of a graph, these characteristics are self-contained by an agent and are treated as if isolated from externalities at this level.

Because individual agents are monolithic systems, they can be decomposed into separate but related groups using similar methods. There are numerous ways to break down an agent, but for this study, the categories are established using the concept of logical and physical architectures as established in the INCOSE Systems Engineering Handbook and the Systems Engineering Body of Knowledge (SEBoK) [33–36].

Both concepts describe ways to model components and relationships within a system. Logical architectures model the functions and subfunctions of a system and the way they interact to dictate the operations of a system, often developed by tools like functional decomposition [37–39]. Physical architectures are the components and subsystems that perform these functions and can often be developed in conjunction with logical architectures. Both concepts cover the external capabilities of an agent and internal operations of an agent, making them a highly useful way to separate characteristics of individual agents. As a result, characteristics are grouped into *Physical* characteristics, describing parameters and concepts related to components and subsystems of a physical architecture, and *Functional* characteristics, describing parameters and concepts related to the function and operations of a logical architecture.

3.2.1 Physical

Physical characteristics, based off components and subsystems of physical architectures, are generally physical properties and descriptions of an agent like mass, speed, and range. These can largely be grouped by the subsystem they directly affect or originate from. Ultimately, parameters within the *Physical* subcategory establish what agents can do and the extremes to which they can be done, often limited by hardware and technology. Many can be described using quantitative and qualitative parameters. Not every agent necessarily contains properties of certain subcategories. For instance, stationary agents lack any movement related parameters, and independent agents may lack communication related parameters. Despite this, the subcategory classification still exists to specify a lack of a property. Common *Physical* subcategories described in this study include:

- Locomotion
- Manipulation
- Sensing
- Transmission
- Processing
- Memory
- Power
- Structure

Depending the agent, more subcategories may be necessary to give proper context to the agent and can be added at the discretion of domain experts. However, most agents typically have properties that fit within these subcategories. These subcategories are described in more detail in the following sections.

Locomotion

Locomotion describes how an agent is able to move through space. A simple way to describe locomotion can be a linear scale ranging from no movement capability to highly mobile. Qualitative parameters would consist of useful parameters describing how motion is realized (wheels, tread, legs, wings, rotors, thrusters, etc.), terrain (land, subsurface, water surface, underwater, air, space, etc.), movement characteristics (omnidirectional, requires forward motion, hover capable), and directional change mechanisms (steering system, control surfaces) amongst other useful characteristics that affect motion. Quantitative parameters within this subcategory include linear and angular speeds, linear and angular accelerations, turn rates and radii, degrees of freedom.

Locomotion characteristics often have a strong impact on *Swarm* level characteristics. Many swarm related tasks require physical motion, which is enabled and limited by the physical characteristics of a swarm's constituent agents. Higher mobility often dictates how fast a swarm can avoid obstacles and accomplish tasks, while movement limits like high turn radii or maximum slope restrict a swarm's mobility.

Manipulation

Manipulation describes how agents can affect the world around them. These are often highly dependent upon the target of manipulation. Characteristics associated with include descriptions of what is being manipulated (shape, size, material), how this manipulation is realized (grabbers and arms, pushers, suction), and the manipulable maximums or minimums, often described in terms of physical properties like geometry, masses, forces, and temperatures.

Manipulation characteristics often focus on external objects, agents, or the environment, and affects the rate at which a swarm performs a task. Maximum weight limits the amount of material an agent can carry, and thus directly impacts the number of trips required to move a given amount of material. Material requirements on the other hand may require adequate locations be found before any progress towards a task is made, extending the time required. Manipulation sometimes overlaps with locomotion as locomotion mechanisms may also be used for manipulation, as in the case of articulated arms and legs.

Sensing

Sensing describes how agents can acquire information from external sources like the environment or external agents, as well as internal conditions. In other words, these are all the parameters related to sensors and their performance. Agents without sensing capability are effectively blind to their surroundings and unable to receive any input from external agents. Qualitative parameters would describe concepts like what inputs the agent is able to receive (electromagnetic signals, visual, audio, pressure, temperature) and appropriate operating environments. Quantitative parameters describe capabilities and limitations of these sensors, like range, sensitivity, accuracy, saturation values, and interference.

Sensing characteristics define what information agents can detect. This correlates to all the types of information a swarm can process as well. In many cases, gathering information is often the primary purpose of a swarm, which is heavily impacted by the characteristics of agents' sensors. Being able to detect other agents is also required for establishing communications and relationships with other agents, critical elements of *Interaction* level characteristics.

Transmission

Transmission describes how agents can transmit information. This, alongside *Sensing*, comprises the two components of communication. Parameters for transmission include the modes of transmission (broadcast, directed), method of transmission (electromagnetic/audio signal, pheromone, motion, body language, displays, exchange of data storage), limitations (terrain, interference) and accompanying quantitative values like bandwidth, operating frequencies, latency, and signal range.

Like *Sensing*, *Transmission* characteristics define all possible types of links can be made to other agents that are elaborated upon in the *Interaction* level. The time required to send a certain amount of information also has impacts on other *Group* level characteristics like *Dynamics*, such as synchronization across large distances or noisy environments.

Processing

Processing describes an agent's capability to process and manipulate information. This often correlates with *Sensing* and *Transmission* as the processing hardware needs to be able to keep up with an agent's information flows. At minimum, agents would be able to perform a preprogrammed list of tasks with no regard to its current situation, either through timed steps or sequential order. Simple processing ability begins to factor in information from sensors to adjust performance, and higher processing would begin to change entire behaviors in response to received information. Qualitative parameters include things like software, operating systems, parallel processing, information complexity and benefits and drawbacks of certain components. Quantitative information would cover processor speeds, number of processors, maximum size of data flows, and efficiency. Processing also considers the minimum processing power for an agent to perform tasks or regulate and control itself.

Processing characteristics are intrinsically linked to *Sensing* and *Transmission* characteristics due to their focus on information. Any of these can drive requirements for the other. These collectively impact *Locomotion* and *Manipulation* as increased processing power often correlates to more mass. *Processing* also directly impacts the *Group* level's *Intelligence* characteristics, establishing what types of swarm intelligence are possible as well as setting hard limits. With other *Group* characteristics, impact is made through the *Interaction* level, where the agent's processing methods and parameters control the relations an agent has with others.

Memory

Memory describes an agent's ability to store and recall information. At minimum agents have enough memory to function, but no ability to store information. Agents that possess memory of some form will need to consider the length of time the memory needs to be stored, data fidelity and redundancy, organization and storage, retrieval, available memory, partitioning, and even data security. These are relatively straightforward to establish for artificial agents, but biological ones are much more complex.

Whereas most of the previous *Physical* characteristics directly address short term concepts like how a task is done, *Memory* characteristics additionally affect operations and long term performance in more significant ways. Parameters like maximum available memory dictates how long an agent can function before this memory needs to be deleted or transferred to somewhere else. Memory is also required for agent learning as well, on both the *Individual* and *Group* levels, especially for the *Intelligence* and complex concepts in the *Dynamics* categories.

Power

Power describes the ability for an agent to perform its functions continuously. Every action an agent takes consumes power, which must be replenished for an agent to continue functioning. Qualitative parameters include the method used to artificially or naturally generate power (internal combustion engine, solar cells, nuclear power, potential energy, eating), their operating environments, benefits, and drawbacks, methods to store this energy (batteries, fuel cells, flywheels, springs, fuel organic molecules), and backup power sources (auxiliary power, afterburners). Quantitative parameters often describe values like energy generation and discharge rates, waste heat, energy efficiencies, amount of power that can be stored, and energy densities.

Power characteristics are similar to *Memory* in the sense that it affects both short term tasks and long term operations. Performing any task requires sufficient power generation and discharge rates to sustain an action, while the duration an agent can perform its mission continuously without pausing to recharge or return is limited by the amount of power that can be stored. This directly affects maximums for agent performance like movement range or communication range which in turn affect the number of trips or length of time an agent needs to complete a mission. When these effects begin interacting with each other, *Group* level effects are directly impacted, especially parameters of the *Dynamics* category.

Structure

Structure describes the way components of an agent are integrated and the way this impacts the agent. Placement and integration of components directly affects the performance, leading to results like optimum coverage or interference from other components. The elements used to integrate these components also affects operating environments, long term performance, and agent lifespan as well. Qualitative parameters include the orientation and placement relations of components, the overall design of the structure, materials used, and operating environments. Quantitative parameters include mechanical and structural parameters (material strength, hardware limits, safety factors, weight) and life-cycle parameters (fatigue cycles, maintenance cycles). Other unique concepts considered here can be the ability for agents to separate or recombine.

Structure characteristics are mostly limited to individual agents and long term effects at the *Group* level, with exception to the *Access* category due to *Structure* dictating an agent's operating environments. Oftentimes, *Structure* places limitations on agents either though physical limitations or integration limitations.

It is also important to note that in cases of virtual code-based agents, *Structure* characteristics are distinctly different from the previously described types of *Physical* characteristics. Whereas all of the other *Physical* characteristics can be represented by specific functions, algorithms, or numerical parameters, the structural characteristics are rooted in not just the parameters modeling physical limitations, but also those inherent to the coding language and paradigm used to design the agent.

3.2.2 Functional

Functional characteristics, based on functions and relations described in logical architectures, tend to be less quantitative and more focused on flow of information and decisions made by agents. As a result, subcategories within the *Functional* category are formed on the basis of what functions and operations are meant to do. Collectively, these functions and their interactions constitute what can be described as an agent's behavior. Unlike *Physical* characteristics, *Functional* characteristics are generally statements of cause and effect with qualifiers. Common *Functional* characteristics are listed below and described in the following sections, but note that this is not exhaustive. Additional types from the perspective of software agents can be seen in Bradshaw (1997) [40].

- Movement
- Information Exchange
- Control
- Disruption
- Tasking
- Beliefs, Desires, Intentions
- Self-Awareness

Movement

Movement describes the logic used to facilitate locomotion and dictate future motion. Tied to the *Physical* characteristic subcategories *Locomotion*, *Sensing*, *Pro-*

cessing, and *Structure*, functions within the *Movement* subcategory determine the specific paths to get to a destination. This would entail details like localized path planning, collision avoidance, and which mechanisms to use at what time.

While *Locomotion* characteristics establish baselines and limits in capability and performance, *Movement* characteristics affect the overall behaviors of a swarm. *Interaction* characteristics often act as modifiers to the movement functions especially in cases where two agents have a movement conflict like a collision and need some way to resolve this, either through one agent's higher position in the hierarchy overriding the other or through some resolution from a third agent.

Information Exchange

Information Exchange establishes when and how to transmit information and what to do afterwards. While the ability to transmit and receive is often determined by parameters within the *Sensing*, *Transmission*, and *Processing* subcategories, *Information Exchange* describes the actual logic and functions controlling to whom and when information is sent. This is dependent on a variety of inputs like internally observed states (goals completed, current location) or externally observed states of others. This also includes deciding what information to share as well, involving aspects of cybersecurity for sensitive information.

Functions and operations dictated by *Information Exchange* characteristics form the basis for many of the links discussed in the *Interaction* level. As a result, they indirectly create the baseline communication network at the *Group* level for completely decentralized swarms. In swarms with more centralization and variations in authority, *Information Exchange* characteristics of an agent can be overridden by a higher authority.

Control

Control describe how an agent regulates itself and its behaviors. Parameters in this subcategory are focused on logic and functions of an agent and how they interact, handling oscillatory behavior, stability, feedback loops, and other features of control. Overall, this describes an agent's ability to maintain internal conditions in order to continue operating, similar to the biological idea of homeostasis. This would handle situations like precedence of certain goals or conflicts among them.

Control characteristics ultimately act as the arbiter between all of the other *Functional* parameters during normal operation. This impacts standard behavior at both the *Interaction* and *Group* levels. These characteristics need to be accounted for when designing any swarm level control algorithms, which often conflict with individual control.

Disruption

Disruption addresses how an agent reacts and adjusts to changes that diverge from standard operation, in contrast to *Control* which addresses standard operation. *Disruption* parameters includes contingencies for disruptions to standard operation (part failure, sudden environmental change, collision). Sudden changes can lead to a complete shift in goals and priorities (immediate return, escape), hardware dependencies (backup power), logic flows (switching methods or algorithms), or entire behaviors (switching from cooperation to independent).

Disruption characteristics by nature alter parameters of the other *Functional* subcategories of an individual agent when triggered by events. These sometimes impact or are caused by changes at the *Interaction* level, such as loss of a pair agent. These disruptions also are a significant component of the *Group* level category *Dynamics*, which accounts for swarm level disruptions in addition to other *Group* level control features.

Tasking

Tasking addresses how an agent prioritizes goals and decides what to do. This is often determines the locations that *Movement* characteristics use for path planning. Specific targets may be better suited to a goal, but goals can also change if a separate target is preferable. The extent that this deliberation is done is one of the key parameters of this subcategory in addition to balancing the agent's maintenance needs.

Whereas many of the other *Functional* subcategories describe the operations that regulate how an agent performs tasks, *Tasking* involves the concept of deliberation where an agent actively chooses to shift its behavior based on decision criteria [3]. At swarm levels, parameters can be overridden by external agents with a higher rank, which in turn imposes more complexity at the *Group* level.

Beliefs, Desires, Intentions

Beliefs, Desires, and Intentions (BDI) are the conceptual drivers behind the logic of agents. Based on the software model established by Michael Bratman in 1987, *BDI* covers a wide variety parameters related to what an agent perceives and thinks, handling what an agent believes is true based on detected information, internalized models, agent goals and purposes, and the ways these are realized [41, 42].

While the other subcategories handle how an agent behaves, *BDI* also addresses why. This is fundamental to establishing *Roles* in the *Interaction* level, which are the building blocks of hierarchies at the *Group* level. *BDI* parameters and the way in impact agent relationships have a profound effect on swarm behavior.

Self-Awareness

Self-awareness describes if an agent is aware of its own abilities and limitations. This includes simple ideas like knowledge of its own physical parameters such that it can plan its own actions without missing a target or colliding with an object, as well as more complex concepts like self-preservation and ability to reassess goals. This can potentially include concepts used to judge other living beings like recognizing itself in a mirror and starts to consider aspects of artificial intelligence (AI).

Self-awareness is similar to *BDI* and has impacts on all three levels. Self-awareness is dependent upon characteristics at the *Individual* level and impact the other two levels in a variety of ways. With sufficient awareness, agents can function independently, requires more resources or complexity to support it.

3.2.3 Examples of at the Individual Level

In this section, a DJI Mavic Air 2 quadcopter and SpaceX Starlink satellite is analyzed using the categories at the *Individual* level. Examining these can be a great way to assess if an agent is appropriate for a swarm application (Mavic Air 2), or to show how agent parameters impact a swarm (Starlink). *Individual* characteristics for a Mavic Air 2 are summarized in tables 3.1 and 3.2 while *Individual* characteristics for Starlink satellites are summarized in tables 3.3 and 3.4.

Based on this information, it can be shown that a DJI Mavic Air 2 is a highly mobile quadcopter able to make point turns and hover but is limited by battery life and weather. It lacks any autonomy besides autopilot, so modifications are required for autonomous operation. It has a good platform for visual sensing but lacks any ability to make decisions on what it sees. As a current off the shelf product, it has great potential in mapping and other visual heavy applications but requires modifications to allow for autonomy and collaboration among agents. Even then, its lack of processing limits its use to applications where planning is done separately and ahead of time, and battery limits require recharging stations.

Meanwhile, the Starlink constellation's requirements ended up dictating the design of each satellite. As satellites that need to maintain a specific orbit, their only locomotion characteristics besides orbital parameters are hall-effect thrusters which

Table 3.1.. Physical Characteristic analysis for a DJI Mavic Air 2 quadcopter [43].

Subcategory	**Characteristics**
Locomotion	Quadcopter mobility; aerial vehicle with limited wind resistance; top speed of 19 m/s
Manipulation	3-axis gimbal system for camera
Sensing	4K resolution camera; forward, backward, and downward facing cameras for piloting; RF receiver
Transmission	1080@p30fps live video feed
Processing	Enough processing power to support autopilot and stabilizing software
Memory	MicroSD card up to 256 GB
Power	LiPo 3s 3500 mAh battery; max flight time of 34 minutes
Structure	Plastic/metal/composite construction; folding frame; 183x253x77 mm dimensions; weighs 570 g

are used for minor orbital adjustments, which can be done autonomously, an important requirement of large constellations. Because their antennas are the primary payload, much of the focus of each satellite is on transmission and receiver hardware, which then dictated the *Processing* and *Power* requirements. Form factor is based on the need to send large amounts of satellites. All other characteristics like *Tasking* are built into the orbit or unnecessary for the satellites role in the constellation.

Table 3.2.. Functional Characteristic analysis for a DJI Mavic Air 2 quadcopter [43].

Subcategory	**Characteristics**
Movement	Hovering and point turns; movement largely dependent upon pilot
Information Exchange	Always active controller signal; toggled video feeds
Control	Autopilot; primarily manually piloted
Disruption	Some stabilizing software for wind
Tasking	Set manually by pilot
BDI	None
Self-Awareness	None

Table 3.3.. Physical Characteristic analysis for a SpaceX Starlink Satellite [44–46].

Subcategory	**Characteristics**
Locomotion	Orbital with Hall-effect thrusters; 550 km altitude, 53 deg inclination
Manipulation	None
Sensing	Optical and radar frequencies
Transmission	Laser linking and phased array from antennas
Processing	On board computers for autonomous debris collision avoidance
Memory	Unknown
Power	Single solar array
Structure	Flat panel design that can be stacked for launches; high albedo for initial satellites

Table 3.4.. Functional Characteristic analysis for a SpaceX Starlink Satellite [44–46].

Subcategory	**Characteristics**
Movement	Primarily locked in orbit; can thrust for orbital adjustments; deorbit capable
Information Exchange	Immediately transmits information that is received
Control	Autonomous station keeping and star tracking based navigation; can be manually controlled from ground
Disruption	Autonomous collision avoidance
Tasking	None
BDI	None
Self-Awareness	None

3.3 Interaction

The *Interaction* level begins to focus on characteristics beyond a single agent, describing the relationships between any two agents in a swarm. These form the building blocks of complex networks and hierarchies at the *Group* level. Like how relationships between individual physical and logical components collectively constitute agent behavior, the relationships between agents collectively constitute swarm behavior. Using the graph analogy, characteristics at this level act as links between nodes within a graph.

Unlike individual agents, relationships at the *Interaction* level cannot be broken down using the idea of physical and logical architectures. Instead, categories are established based on different concepts that impact relationships, namely *Awareness*, *Communication*, and *Roles*. *Awareness* describes what relationships can form between agents, *Communication* describes the relationships that do form and how, and *Roles* describe how relationships are impacted by differences between two agents.

In addition to these three subcategories, other subcategories can exist depending upon the specific type of swarm and its tasks. One such potential subcategory is *Interfacing* for swarms that involve self-assembling agents. In order for agents to attach themselves, they must have the appropriate interfaces to allow this, which can be explored in detail in a separate *Interfacing* subcategory.

3.3.1 Awareness

Awareness describes if an agent is aware of other agents and the environment. This is a critical component for establishing relationships between agents and ultimately dictates the behavior of a swarm. Using the analog to networks, awareness represents directionality of the links between nodes. At its most basic, *Awareness* describes if an agent knows that it is part of a larger entity, if it is aware of its surrounding environment, if it is able to separate swarm members from the environment, and if it is able to transmit and receive information to other agents within this swarm. If an agent knows that it is part of a swarm, that allows the agent's behavior to be directly influenced by other agents or the swarm itself. If an agent is able to identify other agents, then it can begin to factor other agents into its decision-making process. Being able to transmit and receive information to other agents establishes what kind of information can be sent to others and forms a direct link between the two agents.

Awareness is not required for a swarm to function. Unaware agents can still produce emergent behaviors, much like cars in traffic. Being unaware simply means an agent's behavior does not factor in the purposes of other agents into its own decision-making process, only what it observes. Likewise, agents unable to directly communicate and distinguish one another can still form complex swarm behaviors through indirect means like stigmergy, where traces are left in the environment. Being unable to detect other agents simply means an agent is unable to directly communicate and form links. In some cases, especially decentralized or distributed swarms, these are even desired characteristics. Swarms are also able to function without knowledge of its surroundings as well, which leads to simpler agents, but this often leads to poor coordination and adaptive ability.

In addition to these basic characteristics of *Awareness*, there are several other subcategories that address particular aspects of the concept. These are described below and are not exhaustive.

Swarm Awareness

Swarm Awareness describes if an agent is aware that it is part of a swarm and its ability to distinguish other swarm agents from other agents, objects, and the environment. In addition to stating whether a particular agent is able to identify another agent, *Swarm Awareness* also describes the ways this other agent can be perceived and the conditions required for this. This perception can be limited to just seeing another agent within its field of view or can include second-hand knowledge communicated from another agent. *Swarm Awareness* can also be directional: an agent may be able to see another, but not vice versa, while both agents can send and receive signals to each other.

In order for an agent to be aware of others, it must have the requisite *Physical* sensing characteristics to detect signals sent by other agents, whether it be visual, signal-based. Agents must also have the requisite logic and functions to separate these markers from other non-swarm entities. This ultimately dictates what kind of links an agent can form with other agents.

Environmental Awareness

Environmental Awareness describes how the environment affects relationships between two agents. *Environmental Awareness* is the agent's ability to detect and react to these changes such that connections are maintained, or new ones are formed so that the agents can continue to perform their objectives. Some methods of communication are dependent upon certain conditions to function; if these are not met, information cannot be transmitted or received.

The effect of this *Environmental Awareness* is clearest at the swarm level. In cases where swarms are tasked with maintaining a certain geometric formation, an unaware swarm is unable to react to environmental changes like wind, preventing it from reforming broken formations. Swarms that operate in highly controlled, uniform environments likely require less *Environmental Awareness*, but swarms that operate in unknown locations almost certainly require it. This is also a key component of the swarm's ability to adapt to changes as well.

Prediction

If an agent can perceive other agents and the environment, a following question is what an agent does to that knowledge. An possible use of this is *Prediction* which describes how an agent models and predicts other agents. If an agent is aware of and can sense other agents, then it can be beneficial to predict future decisions of these other agents so that it can change its own behavior accordingly. *Prediction* parameters include the particular methods used to model other agents (finite state automata, neural network, response surfaces, decision trees and forests, Turing machine), reliability, robustness, uncertainties, errors, resource demands, and extent to which the methods can be used. Knowledge of the environment can also be factored into prediction models if the computational capability is available. This can also include the ability to avoid collisions or other risky situations.

Prediction is largely dependent upon the *Functional* characteristics and the *Physical* processing parameters, but can be used to create highly complex behaviors at the swarm level. Distribution of predictive capability largely depends on how the swarm's behaviors; highly centralized swarms typically have some agents with high predictive capability to control the swarm, while decentralized swarms have more agents with predictive characteristics, but not to the extent that centralized swarms require.

3.3.2 Communication

Whereas *Awareness* describes whether or not a link can form, *Communication* characteristics focus on the ones that actually do. These are built upon individual capabilities established for each agent in the *Sensing*, *Transmission*, *Processing*, *Information Exchange*, and *BDI* categories at the *Individual* level and are predicated on *Awareness* characteristics between agents. At a minimum, *Communication* depends on if information being transmitted by an agent can be received and interpreted by another. If these characteristics align, then an agent can communicate with another. If the same is true in reverse (not necessarily by the same means), then the agents can establish a two-way link. Depending on the method of transmission, a single transmission can be used to communicate to multiple other agents simultaneously, provided they are all capable of interpreting the signal.

As with *Awareness*, swarms can still function if agents are not able to directly communicate. Information can still be passed indirectly from agent to agent using stigmergy, like leaving trails and pheromones in the environment for others to interpret when they arrive. Swarms that lack any ability to communicate can exist as well, but these tend to be inefficient. Without the ability to communicate, agents are unable to know if a task has been completed, wasting time and energy repeating them. This can be addressed by planning every step for every agent, but this requires extensive planning ahead of time, is inflexible to disruptions, and often counter to the entire reason to use swarms in the first place.

Parameters describing different *Communication* characteristics are grouped into the subcategories described below. Again, additional subcategories can be added depending on the specific type of swarm, but most have these common features.

Range, Bandwidth, and Latency

Range, Bandwidth, and Latency describes the distances and rates that information can be transferred between two agents. Range sets a spatial limit on how far agents can be from each other before communication is impossible and can be impacted by environmental factors. Due to disruptions, there is often a buffer below this maximum to ensure agents do not oscillate in and out of range. Bandwidth sets a limit on the amount of information that can be transmitted of received at the same time and impacts the amount of time agents need to stay in range to receive the full message. The smaller bandwidth between agents dictates the connection's bandwidth. Oftentimes, range and bandwidth are inversely related; the closer two agents are, the higher the bandwidth, provided the hardware is able to keep up. Latency is the amount or time it takes for information to move from one agent to another and is dependent upon the range, environment, and bandwidth available to the agents.

The most significant impact of range is seen in the physical area a swarm can cover (*Access*) and the amount of time for information to propagate or the swarm to react (*Dynamics*). Bandwidth and latency directly impacts *Dynamics* characteristics as well. Overall *Range, Bandwidth, and Latency* affect the swarm's physical coverage and more importantly, operational timescale.

Transmission and Receiving

Transmission describes what other swarm agents a single agent can transmit information to. *Receiving* describes what other swarm agents a single agent can receive information from. Both can be represented in different ways, like a list of "compatible" agents for each agent, or even separate adjacency matrices showing who is able to transmit or receive information across a swarm. Other relevant characteristics would include the method through which information is transferred and limitations to these methods, what type of information is being transferred, and if it is coded or encrypted in some way for security purposes. Common methods are listed below.

- *Broadcast* - single agent transmits information publicly in all directions
- *Directed* - single agent transmits information directly to a specific individual or group of agents
- *Environmental* - single agent embeds information in the environment such that other agents can detect it later

Because swarm agents often move around, it is important to note that actual connections between agents vary over time as well. The number of possible connections will stay constant, but not all of these are realized. Because of this, agents can also include contingencies for creating new lines of communication and reestablishing previous connections. Additional lines of communication can also be established to add redundancy.

Communication often has the largest impact on swarm behavior as much of the swarm's emergence properties depend on communication to exist. These links also form the building blocks of more complex structures seen at the *Group* level, with an entire subcategory dedicated to in within the *Network* category.

3.3.3 Roles

Roles addresses the purpose of an agent among its peers, within the swarm, and the impact of these in relationships. Depending on the purposes of different agents and how they are prioritised, power dynamics emerge. In cases where all agents have the same purpose and role, swarm behavior is largely driven by communication and other links at any given moment. Should there be variance in roles or ranking, the fundamental nature of the swarm changes. With different purposes, some element of cooperation is needed to resolve conflicts, while rankings change the prioritization of agent goals across the swarm.

Roles have a profound impact on swarm behavior at the *Group* level. From a *Network* standpoint, it turns networks into hierarchies with their own distinct properties. This also creates effects seen within *Composition*, which can correlate distributions of power to performance. From a design standpoint, it adds many layers of control that must be considered as well.

Purpose

Purpose characteristics describe the goals of an agent within a swarm. In simple homogeneous swarms, these typically are identical across the swarm agents and matches that of the swarm. In swarms with different types of agents, agents can be specialized, thus creating different roles within the swarm. Some agent purposes can be directly related to the swarm's goal, while other agents serve supporting roles that allow other agents to function, like maintenance, information relay, monitoring, processing, or coordinating agents. Each serve a specific purpose within the swarm and are meant to improve performance.

Purpose for an agent is established by its *Individual* level characteristics and capabilities, and driven specifically by its *BDI*. Differences in purpose are seen directly in the *Group* level category *Composition*, impacting overall swarm behavior. Because there are several concurrent goals, some level of coordination is often required.

Ranking

Ranking describes the power dynamic between two agents. This is often based on the *Purpose* of different agents, but can still exist in homogeneous swarms where one or more agents are granted authority. Agents meant to make decisions and influence other agents possess a “higher rank” than agents that get influenced and create a superior-subordinate relationship. Some agents can depend on other agents for maintenance or information, created another type of superior-subordinate relationship. Agents with similar or identical relationships would share a P2P relationship instead as they do not have authority or incentive to override or influence others.

Relationships due to *Ranking* also vary in strength. Some superior-subordinate relationships can be weak where the influence is small and can be ignored, but some are absolute where subordinates must change their behavior when given an order. In cases where a superior has several subordinates, the strength of this relationship can also vary between subordinates. Subordinates can also have several superiors as well, and should commands conflict, the subordinate would defer to the highest ranked superior or the superior with whom it shares a stronger relationship. If the two are in rank equal and power, then the conflict would need to be resolved by the superiors. Different components of hierarchies based on ranks can be seen in figure 3.3.

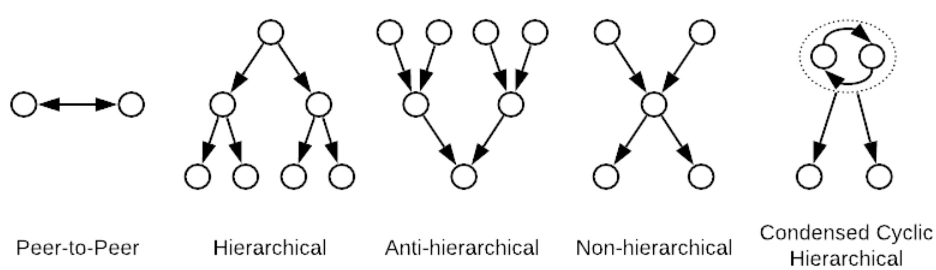

Figure 3.3.. Different Ranking-based relationships between swarm agents [47].

Ranking has one of the largest impacts on *Group* level swarm behavior. It changes the concentration of power between agents in a swarm, directly impacting aspects in *Access*, *Dynamics*, and *Intelligence* and changing the nature of *Network* and *Composition*. Before, any "power" was implicit through network measures like centrality of the agent within the swarm, or number of connections maintained by an agent, but the introduction of *Ranking* can either reinforce inherent power caused location in a network, or shift it elsewhere.

Cooperation

Cooperation describes how agents resolve conflicts. With the addition of differences in purpose or power, there needs to be some way to address conflicts between the two. In cases that involve rank, the consensus typically defers to the superior, but depending on the strength of this relationship, the subordinate can ignore the superior if the command violates its BDI, or if a loophole exists. *Cooperation* characteristics more frequently involve relationships between peers that lack the clear decision priority in superior-subordinate relationships. These can come through prioritization of certain goals or through a metric that determines which command is better.

Cooperation characteristics have less of an impact than differences in *Purpose* or *Rank*, but nonetheless affect behavior at the *Group* level. *Cooperation* often dictates what methods of *Intelligence* are possible, and affect various other *Dynamics* characteristics. It also affects performance, impacting the time for to complete tasks.

3.3.4 Examples of at the Interaction Level

In this section, *Interaction* level characteristics of the PLEXNet exploration architecture simulation studied in [48, 49] and ant colonies [15] is analyzed. Planned relationships between agents in a swarm will be discussed in the PLEXNet example, while the naturally occurring interactions are discussed with the ant colony example. *Interaction* characteristics for PLEXNet are summarized in tables 3.5, 3.6, and 3.7

while *Interaction* characteristics for ant colonies are summarized in tables 3.8, 3.9, and 3.10.

PLEXNet is a swarm architecture consisting of explorers equipped with mapping and research payloads designed to explore a planetary body and hubs equipped with power and communication hardware. Both agent types are mobile, but the architecture is defined by a set of planned relationships. This particular example discusses the virtual implementations in Thai et. al. 2019 and 2020 [48, 49]. In these simulations, all agents are aware of each other and able to communicate with each other with no range or bandwidth limitations and no latency. Agents are aware of their surroundings through cameras, which provided mapping information used to plan future movement. Explorers are dependent on hubs for recharging, data uplink, and access to new locations. Their range is dependent upon battery life and where hubs are, though agents can "petition" for hubs to move to certain locations. Hubs are in constant contact with each other and shared P2P relationships. These characteristics would later define how the architecture functioned. Though this model lacked accuracy to real life scenarios, this shows how certain characteristics in the Taxonomy can be ignored temporarily to isolate behaviors in a controlled environment. Additional characteristics can be added in like latency to improve accuracy.

Some ant colonies often feature many ants with differing roles. As living beings, there are all aware of each other and their environments, but their ability to predict other individuals with precision is unknown. A predominant form of communication between them seem to be stigmergic, using pheromones to coordinate highly decentralized swarms without the need for hierarchies or centralized control. Certain types of ants within a colony have highly specific jobs like queens, which are specific type of ant with unique biology and a dedicated task of mating and laying eggs. Meanwhile, worker ants which form the vast majority of the colony can perform a variety of tasks like foraging, nesting, and building, cooperating when needed to perform complex or monumental tasks. The flexibility of worker ants allows the colony to be flexible and quickly adjust in the wake of big changes. This knowledge, in combination with the

Table 3.5.. Awareness Characteristic analysis for PLEXNet [48, 49].

Subcategory	Characteristics
Swarm Awareness	All agents are aware of each other
Environmental Awareness	Agents are aware of their terrain and resources in the environment
Prediction	Agents are not able to model each other's behavior

Table 3.6.. Communication Characteristic analysis for PLEXNet [48, 49].

Subcategory	Characteristics
Range, Bandwidth, Latency	Not limitation, no delays
Transmission and Receiving	All explorers and hubs can communicate to each other

Taxonomy of Swarm Characteristics can be used to identify other places to explore to better understand ant colonies or derive new behaviors that can be mimicked.

Table 3.7.. Role Characteristic analysis PLEXNet [48, 49].

Subcategory	**Characteristics**
Purpose	Explorers explore unknown locations search for resources on the map; hubs maintain separation and act as recharge and data uplink stations for explorers
Rank	Hubs possess higher rank as they choose locations to move to; explorers follow hubs
Cooperation	Explorers avoid exploring the same location, hubs move so explorers can continue to perform tasks

Table 3.8.. Awareness Characteristic analysis for ant colonies. [15, 50].

Subcategory	**Characteristics**
Swarm Awareness	Ants are aware that they are part of a colony and can distinguish individuals
Environmental Awareness	Ants are aware of their surroundings and act in response
Prediction	Unknown

Table 3.9.. Communication Characteristic analysis for ant colonies. [15, 50].

Subcategory	**Characteristics**
Range, Bandwidth, Latency	Limited by line of sight and pheromone spread
Transmission and Receiving	Ants communicate through physical cues, sound, and pheromones in the environment

Table 3.10.. Role Characteristic analysis for ant colonies. [15, 50].

Subcategory	**Characteristics**
Purpose	Queens mate and lay eggs, drones mate with queens, workers build, forage, fight, or raise young
Rank	Queens fly off to create new colonies, but no true hierarchy
Cooperation	Ants work together to ensure jobs are done in the colony

3.4 Group

The *Group* level consider the effects that arise from interacting agents within a swarm. These describe characteristics that begin to exist with multiple agents, covering simple descriptions of the swarm to complex emergent behaviors. Like how a swarm is a culmination of its constituent agents and interactions, *Group* level characteristics are result of the characteristics from the *Individual* and *Interaction levels*. Returning to the graph analogy once more, this would be represented by an entire graph, or a component of an even larger graph. As most previous studies focused exclusively on a group level, many of the axes and grouping from previous taxonomies can be found in this level.

Because the *Group* level focuses on a single swarm independent of impacts from external agents and swarms, *Group* level characteristics can be likened to *Individual* level characteristics if a swarm is treated as a single entity. As a result, the idea of physical and logical architectures can once more be used to establish categories. However, since much of the physical architecture consists entirely of fully independent, separate agents, which are already described in great detail through the other two levels, a category describing the function and parameters of these agents is unnecessary, aside from one that captures the overall *Capability* and *Composition* of a swarm. The logical architecture on the other hand can be separated into categories that fit within well-established domains of study like *Networks*, *Intelligence*, and *Dynamics*.

3.4.1 Composition

Composition describes many of the observable aggregate features of a swarm independent of links between agents. This contains many commonly described characteristics like swarm size and homogeneity of agents, but also expands to consider homogeneity of other agent characteristics and purposes beyond what kind of agents are in a swarm. This group would also describe the states of agents at any given moment and capture parameters that dictate scalability.

Composition is focused solely on swarm agents, so many *Interaction* level categories do not apply. The exception to this is *Roles*. Despite the focus of *Roles* on links and relationships between agents, it also includes characteristics like purpose of an agent within a swarm, a concept required other agent for comparison.

Size

Size describes the number of agents in a swarm and its impact on swarm behavior. Beyond simply describing the population of a swarm, this can also include number of specific types of agents, agents that have a certain role, or agents within different hierarchies or subcategories. These numbers form the basis of *Homogeneity* within a swarm. At minimum, a swarm will have two agents.

For smaller swarms, the exact number of agents is important. With fewer agents, a single agent accounts for a significant percentage of a swarm and its behavior and outputs. Losing a single agent likewise means a significant loss of capability. For larger swarms however, a single agent has much less of an impact. For a swarm of fifty identical agents, a single agent accounts for 2% of the swarm. If that agent is not critical, losing it would have a negligible on swarm performance. As a result, for larger swarms, a more useful measure of size would be order of magnitude, a measure of *Scaling*. Another way of looking at this is how smaller swarms are dominated by individual behaviors, while larger swarms are dominated by swarm behaviors and aggregate effects.

Homogeneity

Homogeneity describes how similar swarm agents are to each other. Completely homogeneous swarms consist entirely of identical agents with identical purposes and ranks. In cases requiring coordination, democratic methods can be used, or control can be exerted indirectly through the environment or directly from an external controller. Completely heterogeneous swarms have the opposite case, where every agent has different *Individual* characteristics and different *Roles*. These tend to consist a small number of highly specialized agents, often called teams, with little redundancy and are rarely found in large numbers.

Less extreme cases can feature varying levels of homogeneity for different swarm characteristics. A swarm can consist entirely of identical agents but their *Roles* may vary such that different agents specialize in a particular area even if they are capable of more than one. Alternatively, different agent types can fundamentally have the same purpose and operate similarly. In addition, *Homogeneity* can also describe agent states and behaviors at various points during or throughout operation. This is a useful way to observe swarm behavior or diagnose problems by finding correlations between certain levels of homogeneity and performance.

Homogeneity can be represented through proportions or percentages. Different proportions can be used to describe different aspects of a swarm; a swarm can have three types of agents that fundamentally perform two functions, which can give an agent type ratio of 1:3:8 and agent function ratio of 3:9 (or 1:3). An overall swarm ratio can be calculated through several, like treating every combination of agent type and role separately or choosing the most relevant ratio, but the most useful one is largely application dependent.

Homogeneity's impact is seen in overall swarm behavior across other categories, largely through the way it affects *Scaling*. The number of ways it can be calculated is largely dependent upon the the number of agent types and roles. Additional measures can be calculated for other parameters like number of agents with a specific subsystem or component, but whether or not this is useful depends on application.

Scaling

In addition to being a more useful way to describe *Size* for larger swarms, *Scaling* impacts a swarm's overall behavior and ability to grow. As a swarm grows, it generally takes longer for changes and information to propagate. Larger swarms also require more computation power to directly manage. As a result, different scales require different behaviors and parameters to function.

Because swarm behavior is nonlinear, these scales are largely defined by breakpoints. These can be generalized to order of magnitude, but specific breakpoints are largely dependent on the swarm and its available resources. The impact also depends on the *Role* of an agent the the *Homogeneity* of the swarm. For example, adding a second command agent to a swarm that currently only has one other commander fundamentally changes the behavior of the swarm, while adding a second subordinate agent only changes the performance or requirements. The biggest impact comes from three kinds of *Homogeneity*-based changes:

- The addition or removal of agents of a certain type or role that have a small relative population within the swarm.
- A significant change or complete switch in the proportion of agent types.
- The addition or complete removal of a type of agent.

The other critical aspect of *Scaling* is growth. While a swarm stays within its scale, its overall behavior stays the same and the effect of additional or fewer agents is limited to performance and required resources. Should the change cross into another scale however, the change swarm composition can be significant enough to require completely different behaviors to continue operating. Conversely, behavior can limit the scale at which a swarm can operate. Returning to the example with one command agent, a single agent can not have the computing capacity to manage more than ten other agents, requiring additional commanders, and therefore new behavior, to continue functioning. There can be overlap in the scales at which a behavior is effective, however. While the swarm with one command agent can not be able to operate with eleven subordinates, the swarm with two command agents can function, even if suboptimally, with less than ten.

The effect of scaling, while largely driven by *Individual* and *Interaction* level parameters, are only seen at the *Group* level of higher. It fundamentally impacts the number of modes that a swarm needs to operate within and limits the growth a swarm can sustain without requiring more modes of operation. Swarm-based solutions are often cited as being able to scale infinitely, but this is often a vast oversimplification.

3.4.2 Capability

Capability describes what a swarm can do based on its agents. Like individual agents, these are typically expressed in in extremes and are limited by the *Individual* and *Interaction* level characteristics as well as parameters set at the *Group* level. This category primarily describes swarm performance and is based on the aggregate ability of its agents.

This is primarily driven by the *Physical* characteristics of agents, but is limited by relationships in the *Interaction* level and the environment. Unlike the other *Group* level characteristics, its characteristics are results driven and feature concepts most relevant to tasks and objectives explored in the Taxonomy of Swarm Applications.

Access

Access describes the environments that swarm agents can access. These are driven almost entirely by the characteristics and roles of individual agents but can impose requirements on the swarm as a whole. For physical agents this largely covers two concepts: terrain and size. Size describes the locations swarm agents can access despite environmental barriers. Accessible areas include tiny passages and tunnels, dense obstacles, chasms and holes, and other situations where the size and shape is a limiting factor. Terrain on the other hand addresses specific environments agents can operate in. This can be aquatic environments (surface vs. underwater, freshwater vs. saltwater, water motion), aerial environments (altitude, weather conditions), terrestrial (slope, surface vs. underground, biome), space (altitude, inclination, radiation environment), or even specialized urban locations (industrial, urban, populated).

In some cases, having access to multiple environments and scales is desired. Small agents are fast and can enter hard to reach places and give high fidelity information but may be unable to cover large distances quickly. Large agents can carry more equipment and payloads and tend to be more resistant to environmental stresses but may be slow and too heavy to cross fragile terrain. Having a mix allows a swarm to take advantage of both at the cost of more complexity. Having access to different environments also gives information that would have been inaccessible without it or provide mission flexibility.

Increased *Access* is not always useful however. Environmental and size restrictions limit where certain agents can move, which means a certain number of agents need to be set aside to maintain connections to the swarm. Agents capable of operating in multiple environments are also costly and may not be necessary in the environment they are used in, so it is important to balance potential access vs. actual usefulness.

Range

Range describes the maximum physical space a swarm can view or affect without violating its own or its agents' requirements. This is primarily limited by the *Access* and *Network* characteristics of a swarm (and by extension, characteristics from the lower two levels). With no functional limitations imposed by the environment, connectivity, or maintenance, the maximum theoretical range a swarm can cover is the length of the swarm if they form a straight line at the maximum allowable intervals between agents. This however is often not a useful shape for many applications and is affected by the environment. A long chain is also fragile, and can potentially isolate parts of the swarm should an agent fail.

Oftentimes, a swarm instead take on irregular shapes based on environmental restrictions and maintenance ranges. Agents require regular maintenance to either replace broken parts, recharge, or deposit information, which defines the maximum range an individual agent can function around maintenance sites. Swarms of agents are no different. In cases with one maintenance site, the range is effectively a circle centered on the maintenance site for a type of agent. This can be extended through the existence of other maintenance sites or through support vehicles equipped to maintain agents, however. Some of these issues can be circumvented by giving agents the ability to recharge and repair itself, but this requires specific equipment with their own maintenance needs.

Output

Output describes the total output produced from swarms agents. This is generally a simple set of characteristics that can be calculated from the capabilities of agents. For properties than can be aggregated, these can be summed up. For instance, maximum force output can be calculated by summing up the maximum force every agent can exert to move object. Total coverage for some type of information can be calculated this way as well. Meanwhile, other properties are equivalent to the maximum value out of all the agents. The operating life of the entire swarm is exactly that of the longest operational agent.

Swarm *Output* is often affected by the environment. For instance, speed of an aerial swarm is dependent upon wind conditions and the lifespan of a swarm is affected by terrain hostility. Some properties are dependent upon the various *Networks* formed within the swarm. The minimum speed for information to travel from one agent to another is largely dependent on the number of agents that need to be passed and the distance between agents.

3.4.3 Network

A swarm's *Network* is a graph formed from all of the many types of links and relationships within a swarm. A separate network can be formed for every significant type of relationship within a swarm, like communication, physical location, functional dependencies, or control hierarchies. As these networks are formed by agents and their relationships, each network shares the same agents even if they are connected differently. As a result, changes made to one agent can propagate through all networks attached to it. Because small changes can ripple across a swarm through these networks, it is important to characterize the network structure of a swarm in order to understand how an agent or group of agents can impact the rest of the swarm. This resulting "composite network" allows for some networks with multiple unconnected components. This can be seen in figure 3.4.

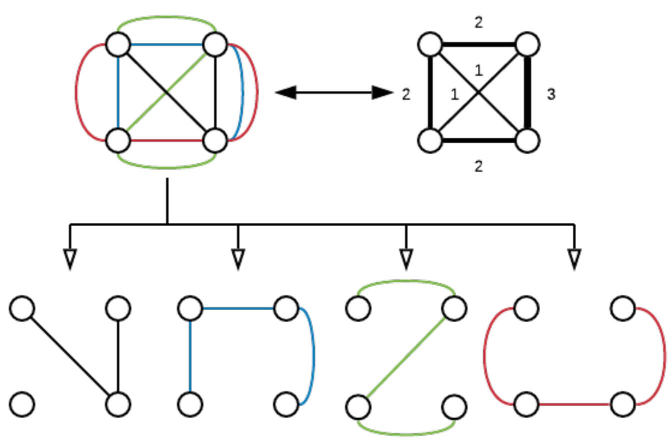

Figure 3.4.. Complete network broken down into separate networks based on link type. Note that the first sub-network (black) does not connect all nodes. These links can also be consolidated to into weighted links (top-right).

Early attempts to describe network structures typically use terms like centralization and distribution, leading to simple classifications like the one presented by Baran (1964), shown in figure 3.5 [51]. Given how similar the ideas of decentralization and distribution are however, Baran's taxonomy has been refined over the years, leading one developed by Maidsafe (2015), where decentralized networks are considered a type of distributed network, which can be seen in figure 3.6 [52]. To better understand the effects of network structure on swarms however, more precise measures are required.

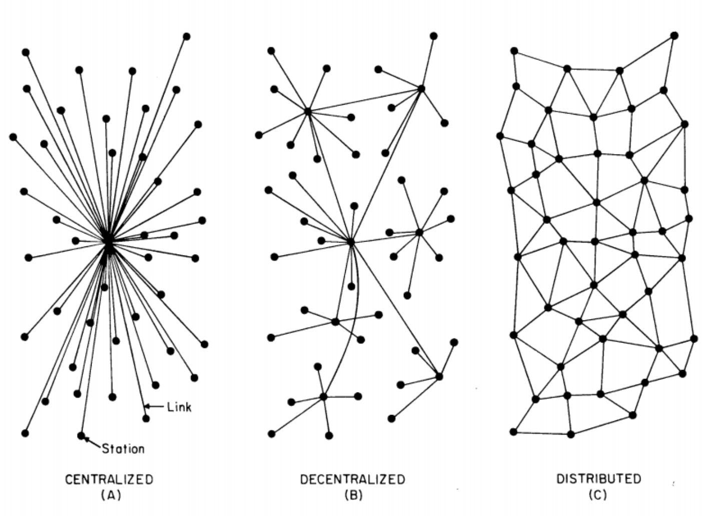

Figure 3.5.. Baran's Classification of Network Types [51]

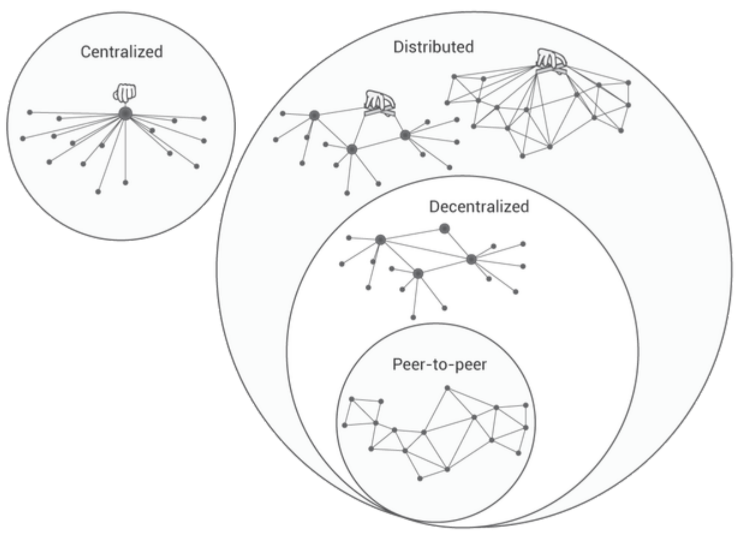

Figure 3.6.. Maidsafe's Classification of Network Types [52]

Many *Networks* characteristics can be observed using metrics employed by network theory [26]. These metrics are well equipped to locate points of various kinds of centrality or describe the connectivity of nodes within a network, as well as the distribution of connections (degree distribution) throughout a network and the tendency for nodes to connect to a certain kind of node. Though generalized, these metrics can show key properties that define swarm behavior, like centralization of a swarm, distribution of control, and robustness and resilience. Meanwhile, network theory methods like Dijkstra's algorithm can be used to calculate the shortest path from one agent to another, which has a large impact on the *Dynamics* of a swarm [53].

This category is divided into two kinds of subcategories. The first subcategories covers characteristics that define the general network features of a swarm using the motivation of Baran's work and network theory: *Connectivity*, *Distribution*, *Centralization*, *Structure*. The last subcategories cover the individual networks formed by a specific type of connection: *Hierarchy*, *Physical Networks*, and *Organization.*

Connectivity

Connectivity describes the number of connections that exist between agents at any given moment. This can usually be quantified by the degree, or number of links attached to a node, of each of the agents. Because these links are often directional, this is divided into in-degree for the number of incoming links, and out-degree for the number of outgoing links for each node. This represents how connected agents are to other agents and is a useful metric for quantifying the impact an agent has. If an agent has a higher out-degree, it has a greater capability to influence others, while an agent with a higher in-degree means it can be influenced by more agents. Note that this can be used to apply to networks formed by a specific type of link, or the composite network of all these links.

Connectivity also has an impact on the number of possible paths within the swarm. The more connected a swarm is, the more options are available to agents when transferring information or performing a task. This also leads to increased redundancy and resistance to failure, as losing one agent is less likely to break critical connections as alternative ones are available. If directionality of links allows for it, more connections also allow for a greater possibility of cycles and loops, critical elements of feedback loops. On the other hand, more links have higher resource demands and comes at the cost of higher complexity. Swarms with fewer links tend to only have a handful of possibilities for swarm actions, making results more predictable.

Assuming that all agents of a swarm are connected, *Connectivity* for a certain type of link can be neatly described on an axis. The maximum connectivity of a swarm corresponds to a fully connected graph, in which every agent is directly connected to all other agents. The minimum connectivity either forms a chain or star pattern in which one agent connects with all others. This can be seen in figure 3.7.

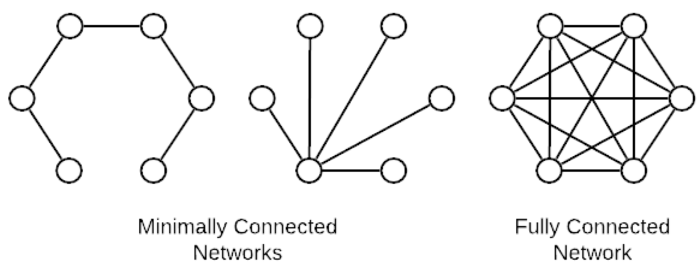

Figure 3.7.. Examples of fully connected and minimally connected networks with six nodes.

Connectivity is often limited by the capabilities of individual agents and the environment they operate in. For *Physically Limited* networks, connectivity depends on the operating ranges of agents and proximity. Connections can only be made when agents are within range of each other. Conversely, connectivity requirements can also impact physical positioning. In cases where a precise number of connections need to be maintained, there are often situations where certain areas become inaccessible to agents because of too few or too many agents in the vicinity. This tends to affect applications based on formation or efficient coverage. For conceptual networks however, *Connectivity* is only affected by the individual agents.

Distribution

Distribution describes how the connections of a swarm are distributed among its agents. Like *Connectivity*, it uses the degree of each agent, but instead of summing them, it plots degree against number of agents to show where links are concentrated. Higher distribution corresponds to more agents with an equal number of links, while less distribution leads to a few agents possessing most of the links. While *Connectivity* allows for higher redundancy and more loops in a network, distribution can impact whether those loops exist. A more distributed network is guaranteed to have fewer critical paths and more loops should *Connectivity* be sufficient to sustain it.

Distribution can be observed in two ways: through the degree distribution of the swarm or through a plot of the degree of each node. For the degree distribution plot, distribution is shown by the difference between the maximum and minimum number of nodes with the same degree (the domain of the degree distribution plot) and the trend in degree as the number of nodes increases. Large differences show disparity in degree distribution, and higher degrees at fewer nodes shows concentration of links towards fewer nodes. Small differences show a similar number of links among nodes. A difference of zero show all nodes have an equal number of links, or maximum and equal distribution. For and individual degrees plot, *Distribution* can be seen from flatness of the distribution. Flatter plots mean higher distributions, while spikes indicate lower distribution (figure 3.8). This is often similar to clustering coefficient, which is based on the number of triplets or cycles of a network.

Though networks featuring low distribution can seem to show some centralizing through its star pattern, like the second example from figure 3.8, *Centrality* is a different concept that low distribution happens to converge to. Overall, the impact of *Distribution* is shown through redundancy, which affects properties like flexibility, resilience, and robustness, and through complexity, a result of having more outcomes.

Using *Distribution* and *Connectivity* as axes, most networks with the same number of nodes generally fall within a "triangle," as shown in figure 3.9. This shape is due

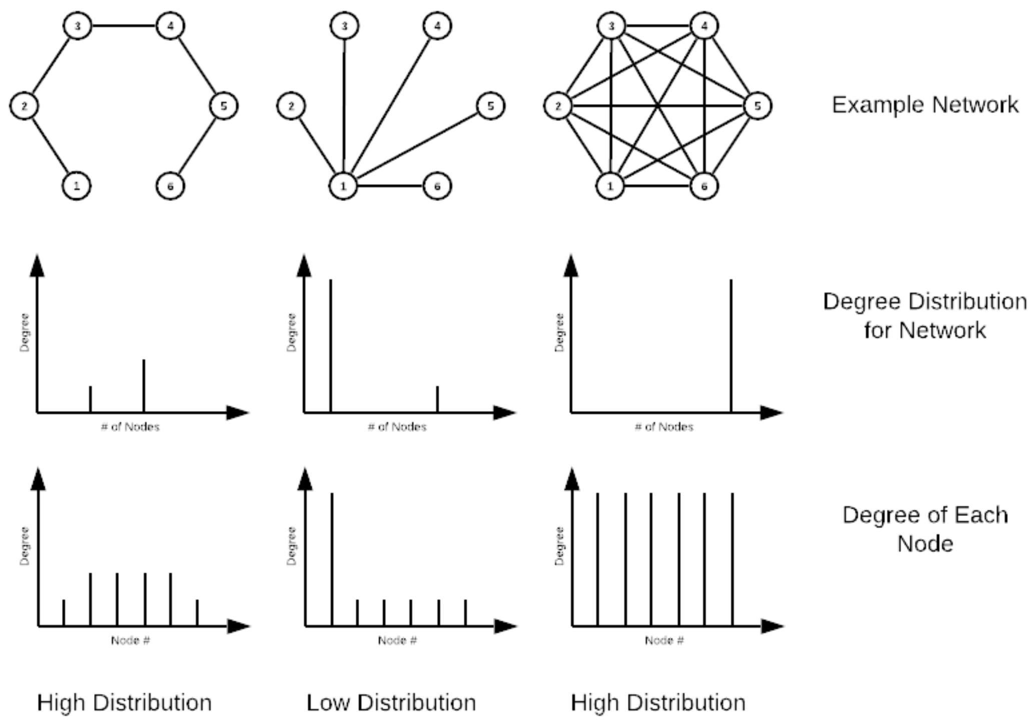

Figure 3.8.. Examples of six agent networks showing distribution of links.

to how *Distribution* increases as connectivity increases as there are a limited number of links that can be made between agents.

Centrality

Centrality describes the criticality of certain agents within a swarm due to placement in the network structure. This is not always dependent on ranking and other hierarchical effects, but almost always impacted by the purposes of swarm agents (as this dictates which links are made). Centrality can be expressed through several ways using network theory (degree centrality, eigenvector centrality, Katz centrality, page

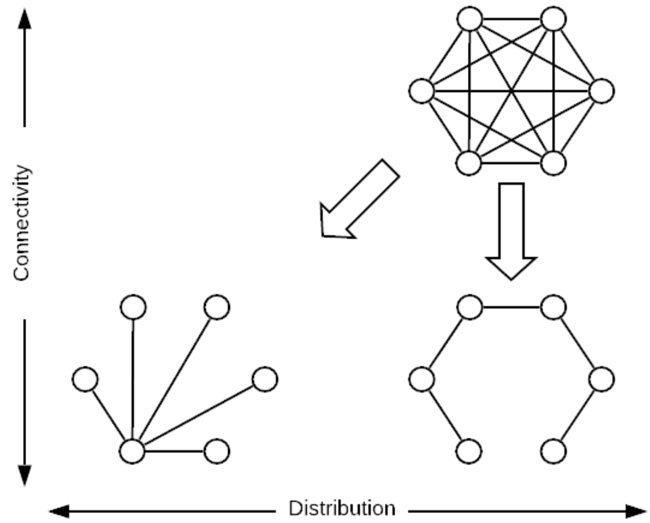

Figure 3.9.. Placement of networks based on Coordination and Distribution

rank, closeness centrality, harmonic centrality, betweenness centrality) [25, 54, 55]. The differences between these measures of centrality can be seen in figure 3.10.

Another way centrality can be expressed in a swarm network is whether or not the loss of an agent results in the overall network fracturing into two or more significant and disconnected parts. These agents tend to have roles that make others highly dependent on them, either for maintenance, accumulation, control. A loss of these would result in a significant loss in function. Networks with these types of nodes often exhibit some form of tree structure where hubs closer to the "trunk" tend to be those with high betweenness, closeness, and eigenvector centrality.

When considering hierarchies and ranks, *Centrality* gains an additional meaning. As hierarchies are inherently a type of network, centrality can be used to describe how many agents possess authority over the rest of the swarm. This can be measured by the number of agents at the highest rank in a swarm. Swarms possessing one top ranked agent would be considered centralized as the chain of command stems from a

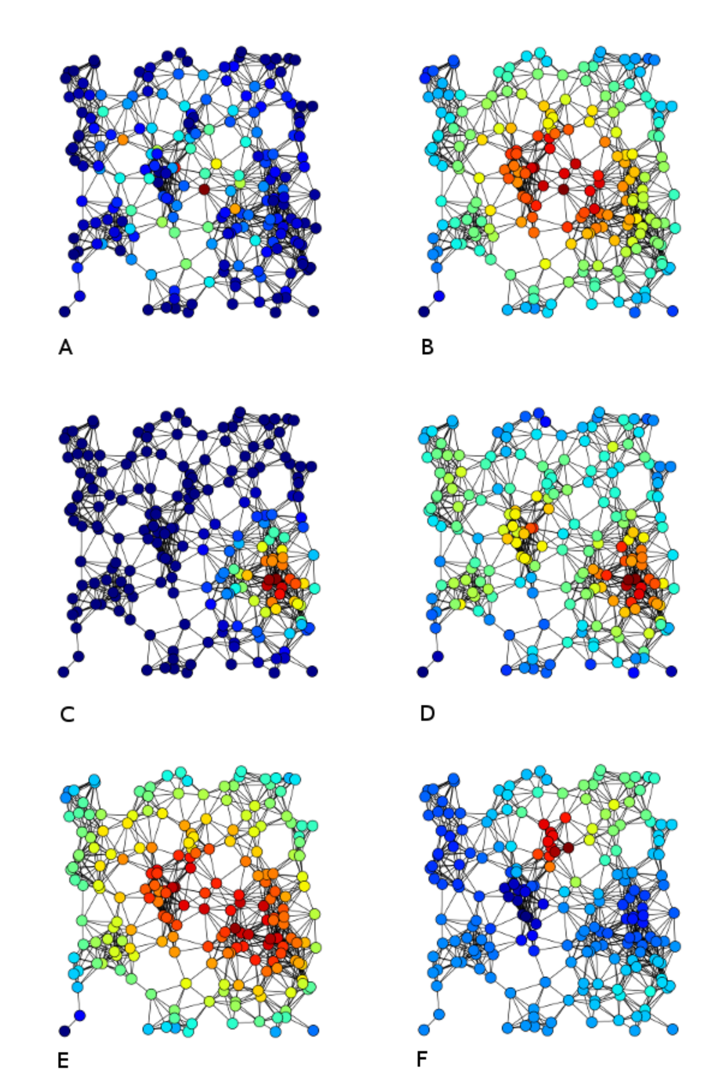

Figure 3.10.. Examples for different measures of centrality for a network, where red indicates more centralized locations. A) Betweenness centrality, B) Closeness centrality, C) Eigenvector centrality, D) Degree centrality, E) Harmonic centrality and F) Katz centrality. [56]

single agent. The more agents that exist on the highest rank, the more decentralized the swarm would be, up to a totally decentralized swarm where all agents exist on one rank. Between these extremes, other considerations emerge like which agents a top ranked agent has authority over, which is discussed in *Consensus*.

These measures are useful for finding points of leverage within the swarm where a small change can impact the entire swarm's state, behavior, or performance. While *Connectivity* and *Distribution* describe the overall connections and structure of a swarm, centrality locates points of influence as a result of these characteristics. Because of the power from positioning within the network alone, this usually has a strong correlation to the functional hierarchy of a network. Given that central locations have such existing power, agents focused on controlling a swarm are best placed in these locations. However, this is not always possible due to environmental or capability based reasons, like the ones mentioned the *Connectivity* section.

Structure

Structure is a culmination of the *Connectivity*, *Distribution*, and *Centrality* characteristics of a swarm, and are often complex structures that cannot be labeled easily. Revisiting the taxonomy from Maidsafe with *Connectivity*, *Distribution*, and *Centrality* shows that simple labels like "centralized," "decentralized," and "distributed" are usually insufficient to describe a network in a meaningful way outside of specific situations (figure 3.11).

Despite this, some archetypal networks do exist and are well studied, allowing them to be good starting places to study the structure of swarm networks. These archetypal networks exhibit a specific level of *Connectivity* and *Distribution* and have associated behaviors that emerge from the resulting network structures. Generally speaking, many of these are based on assortativity, a measure of how nodes tend to connect to other nodes with similar degrees. High assortativity result in dense clusters surrounded by sparsely connected nodes, while low assortativity or disassortativity tends towards hubs and tree structures.

One of these network types are scale free networks, which feature degree distributions that follow a power law [57]. This results in a network where most nodes' sole connection is to a handful of hubs that all connect to each other. This is similar

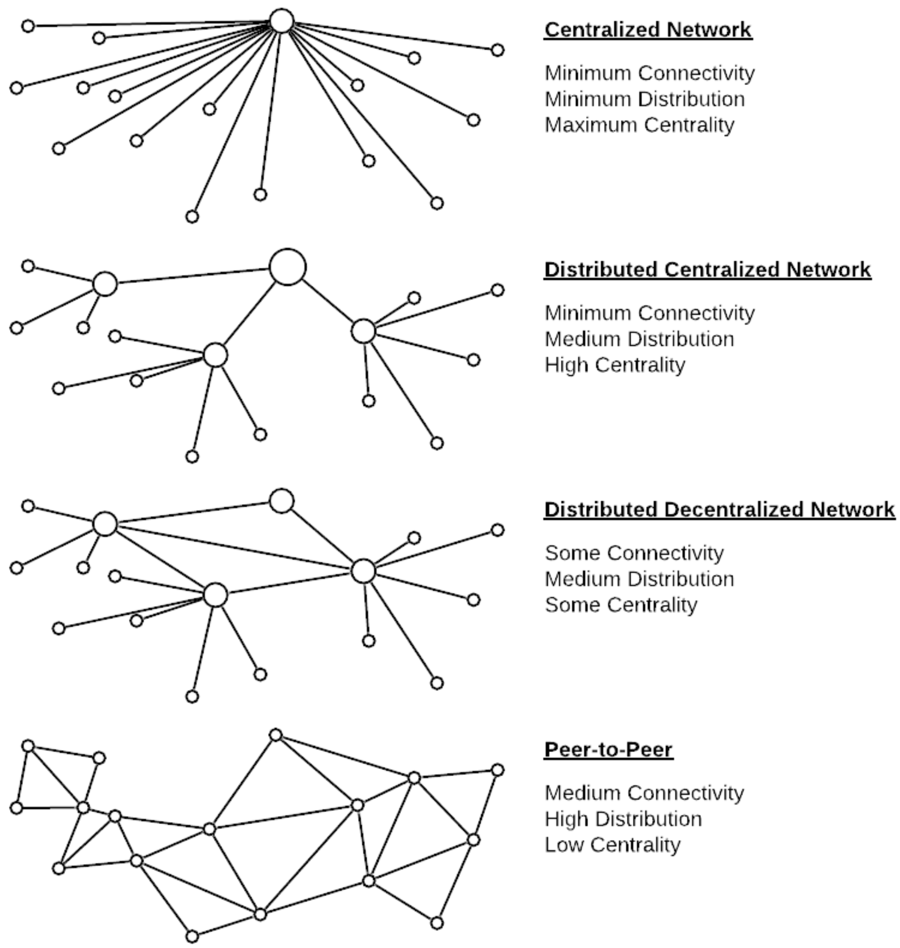

Figure 3.11.. A breakdown of Maidsafe's example networks by *Connectivity*, *Distribution*, and *Centrality* [52]

to networks with a tree structure, or small world networks where most nodes do not connect to each other but can be accessed in a minimal number of steps. Each of these feature many hubs that are critical to maintaining connectivity to all nodes and have their own benefits and drawbacks as a result.

Other types of network structures exist like bipartite networks, consisting of two kinds of nodes that only connect to the other type, or acyclic networks where all links have a single direction and cannot loop back (often the case for hierarchies).

Tree structures can be made more complex by allowing the "leaves" or "branches" to connect. Random networks where degree distribution is shaped like a bell curve also exist and have unique properties like resilience to random failures.

Ultimately, the appropriate network structure for a swarm is largely dependent upon the limitations of the individual agents and the purpose of the swarm. Fewer steps between agents typically result in faster behavior as impacts are felt directly, but as mentioned, this comes at a higher cost and is not always possible. Situations where high predictability or control is desired point towards more centralized systems with lower distribution provided the swarm is small enough to manage. Situations where there are too many agents for centralized control to work likely result in higher distribution and connectivity to create swarms with redundancies and local decision-making. Overall, these effects tend to be in the domain of *Dynamics*.

Hierarchy

While *Centrality* focuses on importance of nodes due to network placement, *Hierarchy* describes the conceptual network of ranks and control within a network. This is created from the rank relationships between all swarm agents and usually possesses levels of equally ranked agents. The agent with the highest authority exists at the top, and each successive level below it denotes which agents have the ability to influence or override the decisions of the agents below it. If multiple agents share a level (including the top level), their influence is either universal to agents below them, or specific to ones assigned by them. In a truly hierarchical network, agents have at most one superior, but this is not a requirement for swarms.

Similar to *Structure* and its various network theory characteristics, this impacts the dynamics of a swarm. The more hierarchical levels that exist in a swarm, the greater the separation between agents and the greater the uncertainty that in the information between the highest and lowest levels. The fewer levels there are, however, the more computation is needed to come to a consensus between agents.

Sometimes, *Hierarchy* matches with the actual network of agents and has little impact on behavior. However, in some cases higher ranking agents are forced to be in less centralized locations due to external requirements like suitable locations for those agents. Typically, the impact of this is a decrease in efficiency. If the aggregation point for the swarm is one of low closeness centrality to the rest of the swarm, then it will take more time for information or resources to aggregate.

Physical Networks

The *Physical Network* is formed by the positions of agents at a given moment. This largely shows whether or not an agent can directly influence another agent, or if their ranges overlap, and is of particular interest for problems involving dispersion or creating formations. If swarm goals do not depend on maintaining positions or connectivity within this network, the *Physical Network* often changes as agents move.

A particular kind of *Physical Network* common to most swarms is a Communication Network, formed from the available lines of communication between agents at a given moment. This largely affects the flow of information across the swarm and is sometimes subject to *Hierarchy* characteristics in situations where this flow needs to be controlled. In this situation, communication is often encrypted or sent in a specific manner such that only agents of higher rank can interpret the information.

Parameters that dictate this network typically affect the shape of the swarm. Lower positional connectivity of agents can result in a stretched out or snaking shape, while higher connectivity leads into rounder shapes. Distribution and centrality would lead to varying agent densities across the environment. Overall structure of this network would effectively impact the areas a swarm is able to quickly respond to. In cases where agents have varying operating environments, the physical structure is often limited by these constraints. Hierarchy can also impact placement as higher-ranking agents may prefer certain locations. Fundamentally, it is defined by the capabilities of individual agents, like power capacity and communication range.

Organization

Organization is the network formed when agents of certain roles are grouped together. From a *Hierarchy* based approach, a swarm can be organized into a strict hierarchy where agents are directly in change of others with no overlap, or a loose hierarchy where an agent can control any agent with a lower rank. From a *Purpose* based approach, agents with similar capability or purpose can be grouped together, leading to groups specialized for one specific purpose, or split into several balanced groups.

Organizational groups can even be established due to physical proximity or dependencies. While some agents can be grouped in such a way that they can self-sustain continuously, this may not always be the case. In this situation, dynamic groups can be established by which agents are in an area performing the same task, or by which agents are currently dependent on each other at any given moment.

Depending on how agents are divided, different dynamics exist. If the organizational structure is based on "implementation, coordination, and control" like in Beer's Viable System Model, these groups need to have firm goals and relationships to prevent one group from overextending [58]. If the organizational structure consists of balanced teams, these teams may end up with little interaction between each other aside from some central authority that provides resources and assignments for the team. Examples of these groupings can be seen in figure 3.12. Overall, these dynamics are similar to their analogs in organizations composed of people.

3.4.4 Intelligence

Intelligence addresses a swarm's ability to collectively reason and make decisions. This category encompasses many of the considerations of artificial intelligence (AI) and swarm intelligence. *Intelligence* typically scales with the complexity of the swarm. In highly centralized swarms where there is a single agent with total authority, swarm intelligence mirrors individual level intelligence. Information from lower ranked sensor

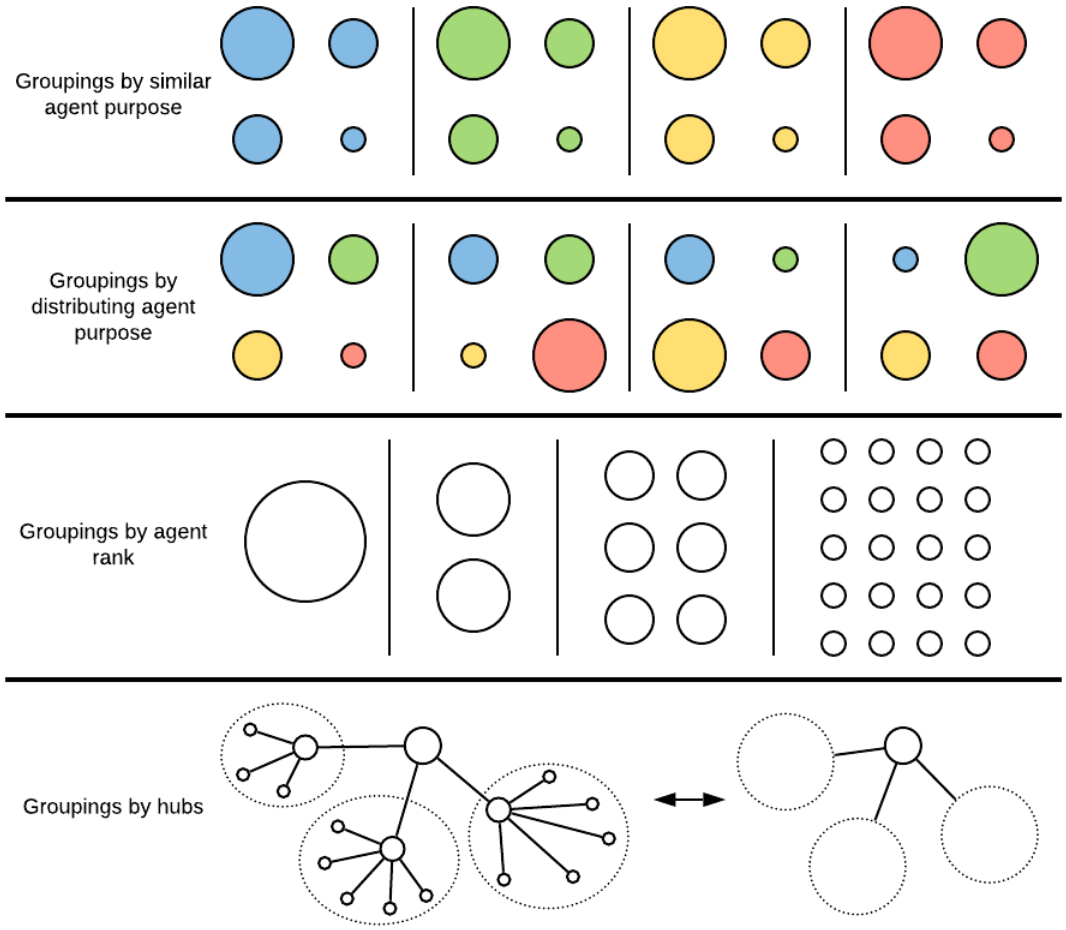

Figure 3.12.. Different ways to group agents within a swarm.

agents is sent up the hierarchy to a top ranked agent issuing commands, much like sensors providing information to a processor issuing commands to subsystems. In this situation, AI focuses on processing information and controlling agents to accomplish the purpose of the swarm. For swarms where there are multiple top ranked agents or no ranking system at all (i.e. higher distribution), *Intelligence* characteristics start to include methods to handle distributed decision making, including problems like conflicting information or goals. Such complexity allows for emergent behaviors like group computation.

Though *Intelligence* covers a vast array of topics, this study focuses on a set of characteristics that are typically seen in swarms operating independently from other purposeful entities, framed around problems at the *Group* level: *Consensus*, *Coordination*, *Group Computation*, and *Learning*

Consensus

Consensus describes how a swarm resolves conflicts between agents. This manifests in two ways, information and commands. Consensus of information addresses how conflicting information from multiple agents is resolved on its way to information aggregation points, including sensor fusion and data fidelity. The goal of this is to ensure accuracy of the swarm's internal model of the environment such that decisions can be made correctly and effectively.

On the other end, consensus of commands or authority addresses how conflicting decisions from multiple agents are resolved. In cases where conflict is among peers of equal rank, this can be addressed through voting (equal votes, weighted votes), deference to relevant agents, or acting separately for strict hierarchies where any agent has at most one superior. In cases where conflict is across ranks, commands from a superior typically override an agent's own decision, but this is not always the case. Exceptions can be granted where violations of BDI can be ignored or renegotiated with the superior. Should there be a conflict in commands from two different superiors, consensus can be made based on the ranks of each superior, consensus between the superiors, the BDI of the agent, or resolved by going up the hierarchy.

Consensus can be a local or global consideration. Highly centralized swarms typically feature global consensus, but distributed or decentralized swarms may be limited to local consensus due to environmental or agent limitations. Local consensus is also a faster as there are fewer agents that need to be considered, but this can risk inaccuracies or more significant problems like dysfunction between groups if not total separation. Consensus may not always be desired, however. Though it is useful to ensure a swarm behaves cohesively, total consensus can lead to useful information being rejected by the swarm. In these cases, local consensus can be more useful that global as some agents do not have the requisite information to give useful insight. This is especially the case in dynamic or noisy environments.

Consensus directly impacts *Coordination* and other elements of intelligence as well as the *Dynamics* of a swarm and is largely driven by the *Purpose*, *Rank*, and *BDI* of agents. The concept has been extensively studied from social and technical perspectives, and methods to accomplish it are often inspired by consensus approaches to human conflicts [59].

Coordination

Coordination addresses how a swarm manages and plans its own moves and manages its agents. While *Consensus* addresses the localized conflicts among agents *Coordination* establishes the context of these conflicts. As agents are sometimes unaware of the overarching swarm goals, coordination often requires agents to behave in suboptimal ways. This manifests differently depending on the hierarchy and network of the swarm but is important to ensure a swarm behaves as intended. Much like how the optimal behavior for a system's components do not mean optimal system performance, optimal agent behavior across a swarm will not necessarily mean optimal swarm performance. As a result, some agents may need to work suboptimally for the sake of the swarm task.

In highly centralized hierarchies, coordination follows the chain of command, in which information is accrued at the highest rank and processed. This information is compared to swarm goals and resources, establishing a set of viable goals for the swarm to pursue. After identifying allocations required for each task, commands can be delegated directly to agents or just to direct subordinates, who then delegate tasks. This depends on limitations of the network; in cases where agents are too far apart or processing is limited, subordinates and middle agents are needed to distribute the

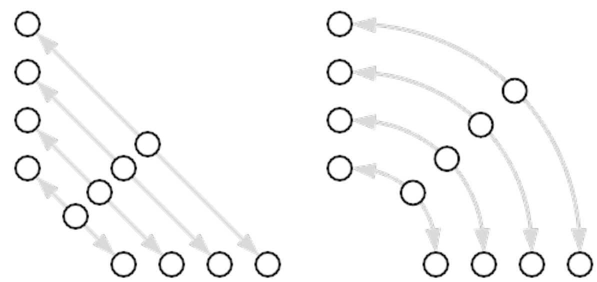

Figure 3.13.. Comparison of swarm motion without coordination (left) and swarm motion with coordination (right). On the left, paths are optimized for individuals, while on the right, agent paths are coordinated to maintain a formation.

tasking requirements. This comes with increased complexity and less certainty on tasking. Assignments can be made randomly or based on ranking criteria like proximity or capability, and some agents may be retasked if the commanders determine another goal would be more beneficial. This type of centralized coordination requires a robust network, high sensing capability, and high processing ability in order to keep commanding agents informed enough to make useful decisions and may be insufficient for highly complex or time sensitive applications.

In less hierarchical systems, coordination may follow the concepts of centralized hierarchical coordination on a local scale where local commanders accrue information and manage agent tasking. Between local groups however, there might be a republic style approach where local commanders collaborate, share information, and vote to make decisions. Depending on the application, consensus may be critical, leading to all groups seeking the same goals. In other cases, local groups may be allowed to choose their preferred goal provided the swarm's goals or resources can allow for it. Alternatively, local groups can compete with each other for resources or goals, where decisions are made based on first-come first-serve basis, high performance metric, or temporary alliances. In this case, local groups seek to maximize certain metrics where coordination is encouraged or discouraged based on the situation. Both of these may involve an additional group that acts as a regulator to break ties or gridlock between groups, but does not actively coordinate groups, rather settling disputes or adjusting the dynamics between groups for better overall performance. As systems become less hierarchical, agents start to directly feature coordination based on cooperation or competition with less centralized delegation.

A third form of coordination can be obtained "virtually" where agents are preprogrammed with certain behaviors and *BDI* that lead to a form of coordination between agents without any dedicated commanders. In a simple case, this may be preprogrammed paths that agents follow to create shapes. In more complex cases, signals given off by other agents may prompt a response from an agent, which can cascade across a swarm. Certain combinations of behaviors can lead to highly complex behavior without any need for a controller, much like how ants can retask themselves based on the pheromone composition left by other ants in the area [15]. This type of coordination is highly complex and depends on a balance of parameters to sustain and may be highly difficult to identify, but genetic and evolutionary methods may be able to identify them. This approach is typically the most sensitive and reactive to changes and needs little computation power to support.

The mechanism for centralization is largely dependent upon the method of *Communication* between agents. Long range communications can allow for highly centralized command, but this may have such a low bandwidth, commands may be limited. Having agents in between allows for more information to be transferred, but this leads to increased costs. If these middle agents and local commanders filter information and delegate tasks, this adds complexity based on the number of agents from one end to another. Scalable methods like pheromones may be able to transmit a high volume of information using just the makeup of a pheromone cocktail, but this is limited to local scales. *Coordination* also must keep *Dynamics* in mind as well. Certain applications may require precise planning and certainty, but this often takes time and is slow to react to rapid changes. Decentralized approaches can handle rapid changes but may be insufficient for long term goals.

Group Computation

Group Computation considers whether or not a swarm functions as a processor or computer. This is heavily dependent upon *Coordination* and the actual processing ability of individuals agents. Individually, agents may act as processors capable of receiving, processing information, and providing results, but collectively, a swarm can act as a supercomputer with as many cores as there are agents, capable of computing large volumes of information. Alternatively, the agents can act as an artificial neural network where each agent is a node [60]. With the many agents available, swarms can be used for a variety of distributed artificial intelligence techniques, possibly creating an intelligent system in the future. As of now, such approaches are best suited to information processing tasks like sensor networks or for filtering information being sent to control agents within the swarm.

Learning

Learning addresses the swarm's ability to take past experiences and use them better address future tasks. This is largely dependent upon the *Memory* and *Processing* characteristics of a swarms agents. Swarms with no learning ability are often reactive, only changing in response to what is immediately occurring. In some cases, this is desirable, but sometimes some learning ability is desired as this gives allows the swarm to predict events before they occur and take appropriate measures in anticipation. This also allows a swarm to function more independently without direct control as agents can resolve conflicts with learned information. Over time, behaviors can even be adjusted to better approach certain problems.

Learning usually affects either the algorithms of a swarm or parameters of agents. Swarm wide changes are often implemented by adjusting constants of an algorithm or even modifying the algorithm itself. Localized changes can be implemented by changing the states of the individual agents, like adjusting movement parameters to lengthen the lifespan of agents. This may result in agents specializing over time, which may lead to higher performing agents at the cost of system robustness. Today, most *Learning* is done using machine learning and artificial intelligence techniques to create a behavior ahead of time before a swarm is deployed, but given sufficient processing and memory, swarms may be able to do this during operation as well. Keep in mind that some learning models may produce false positives or negatives, which needs to be accounted for to make sure the swarm does not make incorrect decisions or learn bad behaviors.

3.4.5 Dynamics

Dynamics addresses temporal features of a swarm's behavior as a result of a swarm's interacting characteristics and parameters. This includes behaviors across many time scales from short fluctuations and movements within the swarm up through long term swarm behavior and growth. Short term behavior is often dominated by local effects and agent parameters while long term behavior is dominated by swarm features and aggregate effects. In a long term perspective, short term behaviors often appear as noise, but these effects may grow into larger effects through feedback loops as easily as they can be attenuated by the rest of the swarm.

These temporal effects can be examined through a variety of lens. Sufficiently large physical distances or operational delays require both *Information Latency* and *Physical Latency* to be considered for synchronization of swarm agents. Control theory is apt for examining some of the short term behaviors by looking at features like *Stability*, and can be combined with system dynamics to examine feedback loops within the swarm network [61]. As a swarm operates over time, its behavior states may shift, leading to *Evolution* within the swarm as agents react to changing conditions in the environment or of their own making. Some of these aggregated patterns can be described using heuristics, but these, along with additional subcategories, are not explored in this study.

Information Latency

Information Latency describes the time delay for information and commands to propagate across a swarm. This is impacted by both the physical distance between agents and the number of steps between agents and affects how synchronized or up to date a swarm is. Low *Information Latency* is highly desired as this means that commanders are always up to date to information available to the swarm and commands can be quickly sent and performed, but the reality is that physical distances and processor speeds lead to delays that must be accounted for in other for a swarm to act as desired.

This delay has two forms: the amount of time it takes for detected information to result in an action, and the time it takes for all agents to be updated. The first form limits the speed at which a swarm can react and is epitomized by the observeorient-decideact (OODA) loop introduced by John Boyd in 1996 [62]. Meanwhile the other form dictates the timescale a swarm can remain synchronized.

If *Information Latency* is not accounted for, different parts of the swarm can potentially work off of conflicting information and cause the swarm parts of the swarm to conflict, or perform tasks based on incorrect and outdated information. Consequences of this can lead to inefficient performance or more serious cases like unusable information or lost agents. The best way to approach this is to keep time sensitive operations local to minimize information latency, while larger timescale operations function globally, but this is not always possible.

Physical Latency

Physical Latency describes the time delay that impacts the physical coordination and synchronization of swarm agents. This is a direct result of *Information Latency* and affects the timing of agents relative to each other when performing tasks. For highly coordinated tasks across large distances, this is a critical characteristic that must be addressed, or else entire swarm functions become impossible to manage. Agents moving before others can may cause some links between agents to break, causing disruptions in the swarm if not accounted for. In some cases, agents may need to move at slower speeds than they are able just to maintain a specific formation, as seen in figure 3.14.

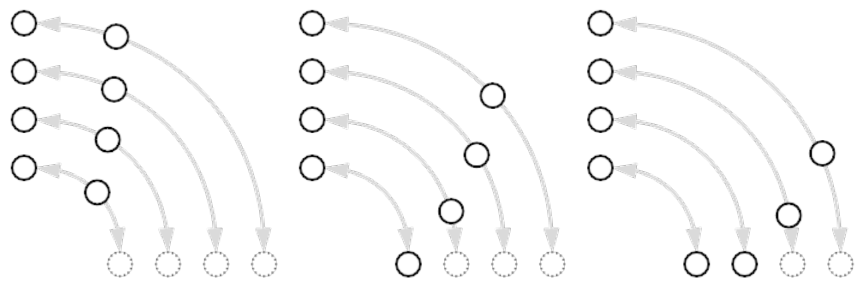

Figure 3.14.. Effect on coordinated motion when physical latency between agents is not considered. Vehicles with shorter paths would need to move slower to maintain formation during rotations.

In swarms where timing is not an issue, this is a nonissue. There may be benefits to synchronization, but without a need, this would just lead to reduced performance. In swarms where synchronization is critical, this requires accurate knowledge of each agent's states, accurate timekeeping, and time for information to be sent, processed, and turned into commands, as well as an additional delay or scheduled time to ensure agents act at the same time. For sufficiently large distances or velocity differentials, relativistic effects may even need to be considered, as in the case for satellites and other space-based applications.

Physical Latency puts a hard limit on the extent that centralized control can be maintained. Should these limits be too great to overcome, a swarm has no choice but to rely on localized or autonomous control in order to function as the *Information Latency* and resulting *Physical Latency* become too great for active control.

Stability

Stability addresses dynamic response characteristics of a swarm as a result of internal or external changes. At its most basic, *Stability* addresses whether or not behaviors or behavioral states converge or diverge over time. Internal commands or external disruptions may cause oscillating behaviors as they propagate across a swarm due to *Informational Latency* or *Physical Latency* and either converge towards stable, meta-stable (oscillatory), or unstable behaviors depending on agent behaviors and relationships. Instability tends to be a result of update rates too high for agents or swarms to handle, leading to undesirable behavior like erratic goal changing. This can be addressed by through slower update rates or trigger-based updates to allow for behaviors to attenuate but slowing the reaction time of the swarm too much prevents it from reacting to changes as quickly. In some applications, some instability may even be desired to break out of stagnant cycles in swarm behavior. An example of this can be seen in the case of ant mills where lost ants start following each other in circles until they die [63]. Though this is a stable behavior, the results are wasteful at best and catastrophic at worst.

In longer time scales, overlapping oscillations between agents or across the swarm may develop feedback mechanisms. Certain combinations of behaviors may lead to a beneficial behavior developing over time as behavioral features begin converging, much like convergence of foraging paths seen in ant colony optimization [14]. Similarly, the interacting agents may cancel out each other's variations over time preventing some instabilities from forming. These largely depend on the size of the swarm and the network structure of the swarm as these dictate the interactions within a swarm. Some of these feedback loops may feature their own oscillatory behavior as well depending on the sustainability of the loops. These feedback loops can be observed in detail using system dynamics [61].

Stability has particular relevance for swarms that involve agents that require maintenance during operation. Moving back and forth to operate and recharge functions as a type of oscillation and some of these behaviors can be interpreted from a stability perspective. Limitations at the maintenance site may impose stability limitations through the number of agents that can be serviced at a time as well. In some cases, these cycles can be synchronized such that a maintenance site is able to operate continuously at a higher capacity, but this is not always possible. These local oscillations may also have impacts at the swarm level as well.

Overall, *Stability* is a tricky category to address from a general level. Many of these effects are highly application specific and depends heavily on all of the characteristics on all levels of the swarm. The best approach to handle the complexity may be to separate the timescales at which loops function or through modeling techniques that can be used to introduce control mechanisms to ensure the swarm behaves as desired.

Evolution

Evolution describes changes in a swarm's behavioral states over time. Though often a result of *Learning*, there are many cases where evolution is a result of the state of the environment or the swarm. In cases without learning, evolution can be observed through sudden shifts in swarm behavior due to progress in goal completion or shifts in *Composition*. As tasks are completed, the environment may be modified to such an extent that different approaches are required in order to make progress. Shifting to this new behavior constitutes an evolution in swarm behavior. For example, search or foraging applications may involve an exploration phase followed by a different phase once targets are found. Mapping applications may feature different algorithms for initial exploration around recharging sites, followed by a different algorithm for longer distances once the local area is fully explored. The swarm as a whole may end up separating into distinct groups as it spreads out to explore.

While natural evolution is a consequence of how agents operate and tend to be more discrete, *Evolution* based on *Learning* is a slower process built on trial and error by agents or the swarm as it aggregates information when performing tasks and slowly modifies parameters over time. This can lead to agents becoming more effective at completing tasks as they identify patterns and use them to predict the future. This is most useful in the development phase when using evolutionary algorithms to develop behaviors fit for use in a final product, or when working in an unknown environment.

Though it address temporal changes, *Evolution* focuses on long term changes across behavioral states whereas *Stability* addresses patterns within a single behavioral state. Though it is a result of agent behaviors, it is largely driven by aggregate swarm behaviors due to time scale and scope.

3.4.6 Examples of at the Group Level

In this section, *Group* level characteristics are explored for the planetary exploration architecture proposed by Quadrilli et. al. 2004 [64]. This explores how some swarm parameters impact behavior and performance of the swarm. Tables summarizing these for each *Group* level category can be seen in tables 3.11, 3.12, 3.13, 3.14, and 3.15.

The architecture proposed by Quadrelli et. al. consists of a herd of mobile "sondes" or deployable sensors that are controlled and deployed by a blimp [64]. The blimp is the central authority in this swarm, deciding where to place sondes and assigning locations for them to move to, making this a highly centralized swarm. The sondes are also able to communicate with each other, but due to limited processing power, are only able to communicate to stay a certain distance away from each other. The network reflects this, where all sondes are linked to the blimp in a hub-spoke shape, but with occasional connections between sondes, as shown in figure 3.15. From a control and intelligence perspective, this is straightforward, where all processing is done at the blimp and commands are sent from the blimp. It is also capable of sensing as well, so in a sense this swarm acts like a distributed monolithic system with the capability for detailed, targeted studies. Synchronization is absent so dynamics are straightforward with agents moving to where they are assigned with slight deviations to maintain spacing. Composition is simple and limited by the lifting capacity of the blimp, while its overall capability is limited by the number of deployable sondes. Different resolutions from the map can be attained by adjusting altitude. The paper also discussing algorithms showing how the swarm can navigate obstacles, while maintaining distances by using artificial physics between sondes [17].

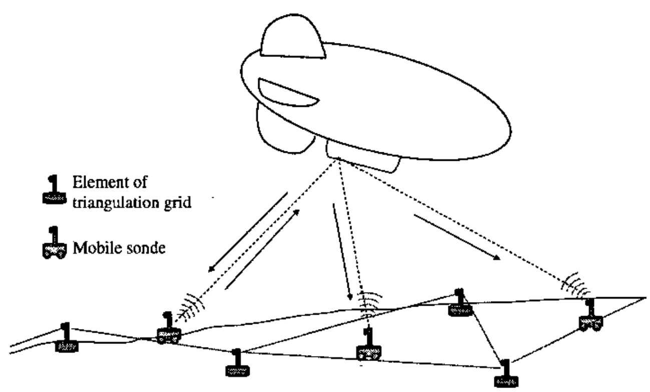

Figure 3.15.. Quadrelli's Blimp based planetary exploration swarm architecture [64].

Using the Taxonomy of Swarm Characteristics, details about the agents can be analyzed from these *Group* level characteristics. Since the sondes are to be deployed by a blimp, they need to be lightweight and small to fit as many as possible. This limitation explains the low communication between sondes as weight needs to be dedicated to sensor payloads instead of communication. As sensor-based agents, they also require some form of memory to store data that cannot be immediately sent back to the blimp. As a planetary exploration system, time delay between a target and

Table 3.11.. Composition Characteristic analysis for the planetary exploration swarm by Quadrelli et. al. [64].

Subcategory	**Characteristics**
Size	One blimp, several navigation beacons, and as many sondes that can be deployed with the blimp
Homogeneity	One blimp vs. 3 navigation beacons vs. the number of sondes
Scaling	Can scale given enough space and some coordination between blimps

Earth makes manual control impossible, requiring the swarm to be fully autonomous. Sondes need some basic intelligence that allows it to traverse unknown terrain, while the central blimp needs a fairly capable set of algorithms to let it perform its purpose. Since the central agent is a blimp, much of the available volume and mass can be dedicated to required computation and communication hardware as it can be inflated during deployment. Because there is one blimp however, that blimp must be highly robust to prevent the entire mission from failing. This comes with additional cost. Overall, this system is simple with well-defined roles, which means that the agents can be simple as well, but the centralized nature of the architecture shows a possible point of catastrophic failure.

Table 3.12.. Capability Characteristic analysis for the planetary exploration swarm by Quadrelli et. al. [64].

Subcategory	**Characteristics**
Access	Blimp functions in air, sondes can be customized for surface or aquatic environments
Range	Infinite range for blimp, sondes limited by terrain
Output	Primarily motion based, with small sampling for research payloads

Table 3.13.. Network Characteristic analysis for the planetary exploration swarm by Quadrelli et. al. [64].

Subcategory	**Characteristics**
Connectivity	Moderately connected; blimps always connected to a set of sondes, sondes connected locally
Distribution	Low distribution, disassortative if there are multiple blimp groups
Centrality	Blimps highly centralized by design, sondes are not
Structure	Network primarily consists of star or hub and spokes where the hubs connect if there are multiple groups
Hierarchy	Two tiered hierarchy where blimp has near absolute command
Physical Networks	Sondes maintain a physical network formed from proximity requirements, blimp stays in range of all sondes
Organization	Strictly follows vehicle type and hierarchy

Table 3.14.. Intelligence Characteristic analysis for the planetary exploration swarm by Quadrelli et. al. [64].

Subcategory	**Characteristics**
Consensus	Less relevant; decisions made by blimps; consensus for multiple blimps not discussed
Coordination	Blimp chooses targets and path planning for sondes; sondes have minimal avoidance and cohesion requirements to prevent collisions
Group Computation	None
Learning	None described; has potential depending on agents

Table 3.15.. Dynamics Characteristic analysis for the planetary exploration swarm by Quadrelli et. al. [64].

Subcategory	**Characteristics**
Information Latency	Latency limited to a few seconds; not as important as swarm does not require synchronization
Physical Latency	Not as important as swarm does not require synchronization
Stability	Unknown, must be simulated; possibly minimal dynamic response features due to control
Evolution	Unknown, must be simulated

3.5 Additional Scopes

In addition to the three scope levels discussed, more levels can be used to address swarm problems. Each progressive level alternates between self-contained entities or relationships between them. Much like how the *Interaction* level considers relationships between *Individual* systems, the level immediately above the *Group* level considers relationships between distinct *Group* level systems. Meanwhile, immediately below the *Individual* level is a level considering the subsystem integration, followed by subsystem, component integration, and component levels. This maps cleanly to the scopes discussed in the ROPE table where entity specific levels map to ROPE levels while relationships address the interfaces between them [23].

For sub-*Individual* level scopes, entire fields already exist to study mechanical, virtual, biological, or social systems, and any analysis should be done using the relevant fields. For levels above the *Group* level however, there is no firmly established approach. There lacks a firmly established broad technical approach to designing swarms, let alone one for addressing interactions of multiple swarms, and technology is only just now becoming capable of supporting swarms. An exception to this would be traffic management, but this is a specific case and cannot be generalized. Despite these limitations, such a scope-based approach can be used to address progressively complex swarm problems. Studying the swarm characteristics can be used to generate possible relationships between swarms, which can be used to build complex networks of interacting swarms. Beyond a certain level of complexity and agent intelligence, concepts from economics, political theory, and social sciences can even be used. Higher levels can also be used to consider the impacts of dynamic environments or external stakeholders. This is an area ripe for study and would be a logical extension to the work in this study.

—

The Taxonomy of Swarm Characteristics is the most comprehensive of its kind and is a critical component of the Swarm Analysis Framework. It provides an extensive overview of general swarm characteristics and parameters, capturing qualitative and quantitative information at multiple scales. It also shows the intrinsic links between many categories, creating a loose map of concepts that can be used to trace dependencies across multiple scopes, potentially identifying causes of emergent behavior. Once characteristics are identified, detailed analysis in relevant fields can be done to develop agents required to create the swarm, alongside control algorithms to manage the entire system. Such a taxonomy can also function as an overview of what a swarm is, providing new designers a large body of knowledge that can help them quickly grasp the ways parameters that influence behavior. The general nature can also connect methods from different fields together, allowing designers to find new methods to solve old problems.

Overall, this taxonomy does not outline methods so much as it outlines capability that can be provided by some types of methods. Specific methods depends entirely on the application and require further research to identify. Despite this, knowing the characteristics of a swarm can greatly narrow down these methods into more appropriate ones. This can be used in conjunction with the Taxonomy of Swarm Applications to better understand a swarm and its purpose, making their design a much less daunting task.

4. SWARM ANALYSIS FRAMEWORK – SWARM APPLICATIONS

Because swarm design is primarily driven by swarm applications, it is also useful to develop a tool to organize and understand swarm applications. Bayindir [5, 17], Mohan and Ponnambalam [18], Navarro and Matia [19], and Balch [20] have all provided different ways to group swarm applications, but while they are useful for linking methods, metrics, and applications together, their organizational structure makes adding other swarm tasks like the ones introduced by Arquilla and Ronfeldt [8], Clough [4], and Scharre [21] unwieldy. Generalizing to all types of swarms would simply lead to the taxonomy growing until it becomes too large to be useful.

Instead, this section proposes a different way for organizing swarm applications. Instead of treating each application as separate problems, it is useful to take these applications and break them down into different components, which can be combined to create simple or complex applications. These components are also much easier to organize as there are a limited number of them, in contrast to the theoretically infinite distinct applications that can be made from these components.

These components predominantly fall into two distinct groups: *Tasks* and *Objectives*. *Tasks* specifically describe what a swarm is supposed to do, while *Objectives* describe abstract principles and goals that impact swarm performance. *Tasks* are mutually exclusive for a given set of agents, have different goals requirements, and are generally focused on end results. In contrast, many different *Objectives* impact a given set of agents at the same time and have principles that generally remain constant regardless of the *Task* they are meant to accomplish.

Note that this breakdown does not capture every application proposed by research, like coordination, tasking, or adaptation. This is because these concepts and methods are directly addressed at the *Characteristic* perspective of a swarm. These are not goals so much as they are internal operations meant to allow a swarm to complete tasks, and would be better addressed from a capability perspective than an performance perspective.

The next two sections describe different types of *Tasks* and *Objectives*, and show how the applications described in past studies fit within this taxonomy. Following this is a section that describes and breaks down aeronautical, astronautical, and mixed domain applications.

4.1 Swarm Tasks

Tasks are the basic activities that agents are meant to accomplish and can be combined to created more complex tasks and applications. These refer to actions that require multiple agents, as opposed to actions that can be accomplished by a single agent. They can largely be group into four categories inspired by the *Group Movement* axis from Balch's taxonomy for tasks [20]: *Aggregation*, *Dispersion*, *Formation*, and *Search*. *Aggregation*, *Dispersion*, and *Formation* all describe goals where agent states converge, diverge, or maintain some specific pattern, respectively. Generally, this applies to physical location, but this can be abstracted to address tasking, coverage, or many other concepts. *Search* is a unique case where the goal is not some trend or pattern, but a specific target.

All four of these *Task* types can focus on either external objects or swarm agents, matching the *Subject of action* axis from Balch's taxonomy for tasks [20]. A focus on external objects typically involves swarm agents changing the state of objects to match the task, while a focus on swarm agents involves the agents changing their own states to reach the goal set by the task. Functionally, the goals have similar concepts, though the *Objectives* required can be vastly different. While *Group Movement* and *Subject of action* axes from Balch's taxonomy are useful in creating the categories

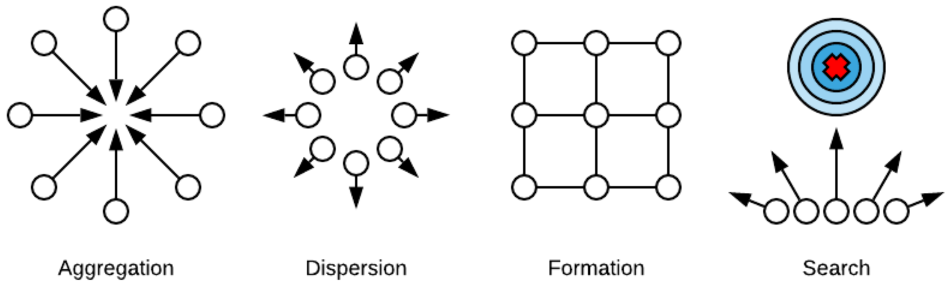

Figure 4.1.. The four basic *Tasks* that swarms are capable of performing.

within *Tasks*, the other axes addressed limitations and metrics, which are not useful when identifying distinct goals from a general standpoint.

This section explores each of the four *Task* types and map some of the applications described in previous studies into the most relevant types. Though the four types are meant to be comprehensive and distinct, there is some overlap between types. *Formation* can be understood as a balance of *Aggregation* and *Dispersion* and *Search* can involve features of the other three, so some specific tasks can be hard to place. The specific tasks listed under each *Task* type are also meant to show what tasks share a common goal, and are not exhaustive. Many of the examined tasks and their associated methods are based in Bayindir's analysis [17].

4.1.1 Aggregation

Aggregation tasks are those whose goal is to adjust the states of agents or objects such that they converge to specific values. In cases where agents or states cannot overlap, a minimum spacing is enforced. Oftentimes, these tasks involve gathering agents and objects into a single or multiple locations. *Aggregation* has many uses. Swarms that can gather towards a spot at the end of a mission make it easier to pick up the agents for storage without needing to track them all down, and having agents or objects in one spot is good preparation for something to connect or affect them at one instead of several times in different locations. *Aggregation* also allows for agents to pool together their capabilities when in the same location to accomplish monumental tasks while remaining simple. This pooling also provides some redundancy to when accomplishing a task but comes at a higher risk for losing more agents to a single disruption.

Gathering generally involves two key concepts: where to gather and what to gather. Gathering points can be absolute like some external coordinate or the location of a specific agent or can be relative based on what a swarm knows, like the centroid of the swarm or some calculated location. Locations can also be determined by the swarm based on criteria like appropriateness for the agents or objects being gathered, or separation and proximity to other sites for cases with multiple gathering points.

What to gather first considers whether the subject of the task are the agents or external objects. If the subject is an agent, then *Aggregation* usually involves agents converging to a predetermined or calculated spot, with self-segregation in cases with multiple spots based on proximity or type of agent. If the subject is an object, the agent needs to locate an object (possibly involving elements from *Search* if the locations are unknown or constantly changing) and physically move it or adjust its state. In both cases, rules posed by the environment like terrain impact the specific path to the aggregation site.

Considerations with *Aggregation* typically involves either some collision avoidance logic if not some form of path planning and formal *Coordination*. This is largely dependent on how much individual agents can process and the network structure of the swarm. Metrics that have traditionally been used in studies either involve time to complete the task, or if the swarm is able to successfully aggregate. In past research, there are several applications that prominently focus on aggregation. These are described below.

Gathering

Gathering is the simple task of getting agents or objects to a single location or state, and acts as the basic form of *Aggregation*. As described from above, methods to determine where and how to gather can vary from a predetermined absolute location or agent to one calculated based on some metric like location. Other methods can be really dynamic like those using artificial physics where agents move based on "forces" exerted by other agents like cohesion, connectivity, or separation. Methods using artificial physics are not exact and are usually chaotic, but for sufficiently large swarms, chaotic agent movement gets dominated by the forces acting on the swarm. Certain combinations can even create unique results, like agents clumping into local clusters which then clump towards each other. Locations can also be determined using cooperative methods where agents come up with candidate locations and vote.

Clustering

Clustering is a more complex form of Gathering that involves multiple locations. As a result, this may require some way for the agents to classify and separate themselves or objects. Locations can be chosen using the same methods as for Gathering, with the only difference being how entities are sorted and requirements like spacing of the clusters. If the task is to simply go to the nearest location, this would essentially create the clumping example from Gathering, but without the need for the clumps to gather. If there is some need to sort and gather objects, then agents would need to be equipped to detect key differences in objects in order to sort them properly.

Flocking

Flocking is simply the dynamic version of Gathering or Clustering where the aggregation point moves. This can have an absolute form where agents move towards some predetermined path or follow a designated leader that controls where the swarm goes, as well as a probabilistic form based on artificial physics like Craig Reynold's Boids [16]. In addition to using position based artificial physics for Gathering and Clustering, Flocking usually needs to take speed and direction into account to create a flock. Multiple flocks have additional parameters to determine which agents form which flocks and make sure separate flocks do not collide, or in cases where they do, ways to avoid individual collisions. Depending on how the flocks are formed, some agents may switch flocks.

Collaborative Manipulation

Collaborative Manipulation usually involve a heavy or unwieldy object exceeding the capability of single agents or involves a mechanism that requiring multiple agents to operate. Collaborative Manipulation involves two phases: gathering agents to the object and coordinating agents to manipulate the object. Agents can be gathered with any of the methods described for Gathering and Clustering, but for cases where this is not a primary task, it may take time for enough agents to actively choose to divert from their original task to help with the Monumental Task. Coordination largely depends on what is being manipulated.

Overwhelm

Overwhelming is accomplished through the number of agents alone. Overwhelming is often used in more adversarial applications like sending more agents than a threat can handle simultaneously, like bees swarming a hornet to raise the temperature beyond what a hornet can survive [65]. Overwhelming can also be used in less overtly aggressive applications like creating a large dynamic cloud that can overwhelm sensors and provide cover for other agents [4].

4.1.2 Dispersion

Dispersion tasks are those whose goal is to adjust the states of agents and objects such that they diverge, with some limitation as to how far they can diverge. This is essentially the opposite of *Aggregation*. Whereas *Aggregation* accomplishes monumental tasks by pooling the force exerted by agents hardware and *Manipulation* characteristics, *Dispersion* accomplishes monumental tasks by limiting the amount of overlap between agent capability. Maximizing distances between agents can often be used to maximize swarm coverage and reducing overlap between agent capabilities and states can allow for highly capable swarms at the cost of some fragility in terms of overall application, but resilience to local disruptions.

Like *Aggregation*, *Dispersion* has two key concepts: who disperses, and in what domain. In some cases, often homogeneous swarms with simple purposes, the entire swarm is meant to disperse. Other times, only certain agents are meant to disperse from each other, so while a set of agents are far apart, they may be close to other agents in their swarm. This can occur on multiple levels where some agents are a certain distance apart from each other, while the remain agents are meant to keep a smaller distance away from these specific agents as well as each other, creating a structure of connected hubs.

Domain addresses what states are meant to be separated. Most often, this refers to location, but this can include separation in 2 or 3 dimensions, a temporal axis, and environmental features like terrain or weather. This can also be conceptual as well where agents have a dispersion of function. Dispersion of specialties allows a swarm to do more things, while dispersion based on targets allows a swarm to sample many things at once. Sensor heavy applications tend to involve *Dispersion* tasks as a result.

Considerations with *Dispersion* usually involves a combination of *Optimization* and *Coordination*. Ranges are determined by *Interaction* characteristics of a swarm and parameters are largely depending on the specific task. Metrics are usually based on optimization and efficiency, as well as time to disperse. Due to limitations based on connectivity, some *Dispersion* tasks have overlap with *Formation*, but *Dispersion* tasks are fundamentally about maximizing separation of agents with less regard for any particular pattern. Prominent examples in literature are described below.

Coverage

Coverage is the basic dispersion problem where a swarm or group of swarm agents separate to maximize the distance between each other. Regardless of which agents are meant to disperse, the principle for coverage is to maximize the amount of information that a group of agents can cover. This can apply in a single axis, like ensuring that the swarm collectively has to ability to detect frequencies across the electromagnetic spectrum, or with coupled axes like x, y, and z components for physical location. This does not mean that this coverage needs to be fully continuous, but this would likely be desired. Spacing between agents are heavily dependent upon the range they can cover and varies if agents have different sensor ranges. For physical locations, there is almost certainly overlap as ranges function as radii, creating circles and spheres of coverage. In practice, coverage is usually planned ahead of time or involves some active centralized coordination, but it can also be accomplished using probabilistic methods where agents avoid each other using a separation parameter, with some limitations like minimum number of connections and maximum distances. For location-based problems, the final shape of the swarm largely depends on the algorithm and where agents are to begin with. As coverage often relates to sensors, an additional consideration is the resolution of sensors based on distance. This is especially relevant for aerial swarms where altitude dictates resolution.

Exploration and Mapping

Exploration takes the concept of Coverage and adds a temporal dimension to it. Not only are agents supposed to avoid each other physically, they are also supposed to avoid the areas previously covered by themselves and others. This task is necessary when there are not enough agents to support Coverage for a large region, so agents need to move as a result. Exploration and Mapping demands agents with high quality sensors, durability, mobility, and memory as well as enough processing power to manage coordination if no central agent exists. Methods to do this include the same ones for Coverage like initial planning, coordination, or artificial physics with an emphasis on separation (in contrast to convergence in flocking). Agents can be programmed to make one of several movement patterns, or they can deliberate and choose on their own. Specified paths tend to be optimized for exploration, but this may be ineffective and inflexible when faced with turbulent environments or difficult terrain. Localized deliberation is much more flexible but not guaranteed to be optimal. This type of task assumes the target of exploration is relatively static over time.

Patrolling

Patrolling adds further complexity to Exploration and Mapping by considering a time varying environment as well. It is a useful way to maintain Coverage over a region where there are not enough agents to support static coverage. Whereas passing by an area or state is sufficient for Exploration, a dynamic environment or target requires agents to revisit an area to update the previous model. Methods to accomplish this are similar to Coverage and Exploration, but with the added requirement that agents return to previous areas within a certain time frame. In general, this can be done by turning agent paths into repeatable cycles, but it can also be accomplished using probabilistic methods assuming parameters are tuned well.

Fractionated Systems and Teams

Fractionated systems take the idea of dispersion and apply it purpose in addition to coverage. Fractionated systems are based on taking a monolithic system and dividing its functions across several agent such that all component agents are necessary for the system to function. Resilience can be factored in by having duplicate agents so that loss of a single agent does not cripple the system, but overall, functions are divided across the swarm. This type of problem is of particular interest in satellite systems where launch mass is an absolute premium. Allowing a single high capability communication satellite to be shared by several nearby satellites allows these satellites to be much simpler and lighter, only requiring communication systems that can reach this main communication satellite instead of ground-based antenna. Conceptually, this division of labor is also implemented in teams of people where personnel is limited. In these cases, it is more important to distribute functions across the team as each agent has high resource costs.

4.1.3 Formation

Formation tasks are those whose goals involve adjusting the states of agents or objects such that they create a specific pattern or shape. This falls between *Aggregation* and *Dispersion* in the sense that the goal is not for agents to get as close to or far away from each other, but with the additional condition of creating a formation as well. Like these other two tasks, it can involve parameters like which agents and objects are intended to create a formation and where this formation is meant to be created, but the key parameter is the type of formation being formed. Specific types of formations have vastly different methods to accomplish them and different applications they are best equipped for. Depending on the application and methods, formations can be fluid and loosely enforced, highly controlled and ordered, or automatically generated based on a set of parameters.

Many swarm related applications involve some aspect of *Formation*. *Aggregation* and *Dispersion* are extremes focused on convergence or divergence and usually do not provide much nuance that some problems require. This nuance comes with an incredibly diverse set of problems that have been studied extensively. Formations can be used for a variety of different applications and being able to create specific types of networks or shapes autonomously can be highly useful for conveying information or setting up robust systems. This takes full advantage of the options available at the *Interaction* level to generate effects at the *Group* level.

Since *Formation* covers such a large area, considerations and metrics largely depend on the specific application. Some of the more prominent examples in research are described below, along with specific methods and considerations.

Relative Positioning

Relative positioning is a loose type of *Formation* where the goal is for agents or objects to maintain some distance away from each other. This is functionally the same as a Gathering or Coverage task, but with spacing somewhere between the minimum or maximum allowable distances. This spacing is often a specified range but can vary as the application requires. This is also the loosest interpretation of formation and fits neatly in the spectrum between *Aggregation* and *Dispersion*, using identical methods to achieve.

Connectivity

Connectivity requirements adds some complexity to Relative positioning by making sure agents maintain a certain number of links to each other. This is can be achieved manually ahead of time, but most methods achieve this autonomously by having agents checking with each other periodically as the move. Some buffer is usually built in to prevent agents from losing track of each other. The goal is just to maintain a certain number of connections to balance resilience to agent losses and cost of maintaining connections. Because connectivity is the driving factor, the resulting formation is loose and can be chaotic, but increased connections can end up creating rigid shapes by virtue of too many restrictions. Requiring only one connection can lead to disconnected pairs, while two connections start to form chains and rings. Three or four connections can lead to irregular triangular or square tiling in the swarm but are still flexible enough to move around. Five or higher links per agent start to make clumps with limited mobility.

Patterns and Shapes

Patterns and shapes tasks require agents or objects to be in highly rigid formations that are functionally the opposite of formations created through connectivity. This encompasses problems like having agents create geometric shapes, polygons, and curves, or continuous patterns like zigzags, grids, and lattices. This is a highly specific problem that requires that agents be able to accurate keep track of their own position as well as the positions of others. This can be achieved through some central coordinator that plans the motion of other agents such that they can create a shape, but most problems have focused on how to generate these shapes autonomously using local information. These problems are tricky, requiring highly accurate information and an algorithm to use that information effectively.

Methods to accomplish this are usually based in math and geometric relationships. Linear patterns like grids and lines are easier to generate as the mathematical principles to create them are simpler. Curves on the other hand are much harder. One famously difficult problem is circle formation where agents attempt to form an evenly spaced circle with themselves or objects but fail due the limitation of the algorithm or sensors [66]. A radius focused approach can just as easily lead to a Reuleaux triangle as it can a circle. The problem becomes much more complex when the goal is not a simple geometric shape but a highly unusual shape that cannot be mathematically generated easily, but this has been accomplished successfully [67].

Because of the many types of shapes that can be formed, the most appropriate methods depends on what shape is being formed. Path planning is a highly important consideration to prevent collisions or cases where agents need to make obscenely long moves to the other side of the swarm. This is even more important if agents need to locate and move objects into place.

Network Building

Network building is a more advanced form of connectivity based *Formation* where connections are made based on a certain ruleset. The number of connections that can be made varies based on the condition of the swarm or the environment, which can be used to create highly elaborate networks. Limiting possible links to those of adjacent ranks can lead to a network based on swarm hierarchy, while limiting agents to those with different purposes can be used to avoid repetition and redundancy locally. Though the results are probabilistic, similar structures arise in most iterations as an emergent behavior of the ruleset. Since the formations are networks, they can often take advantage of network growth and expansion models like the Barabási-Albert model for generating scale-free networks [68].

Coordinated Motion

Coordinated motion takes any of the three previously described types of formations and adds movement. In this task, the goal is to either maintain a specific formation as it undergoes translation and rotation or manipulate agent paths such that transitions from one shape to another are not messy. Agents all have strict relative positioning requirements as they move to preserve the shape and make smooth transitions, and their paths can vary depending upon the extent of a transformation. Small rotations can allow linear movement, but larger rotations require curves to keep the shape from shrinking during the transition, or multiple steps in between to simulate curved paths. Path lengths also vary between agents, necessitating different velocities between agents. This is especially true for rotations: agents farther away from the point of rotation must travel faster in order to maintain the same angular velocity as other agents much closer to this point. Coordinated motion for Patterns and Shapes have the strictest requirements as the entire purpose is to create shapes. Network or Connectivity based formations on the other hand are more flexible, but still have limitations to prevent the fundamental structure of the formation from changing during motion.

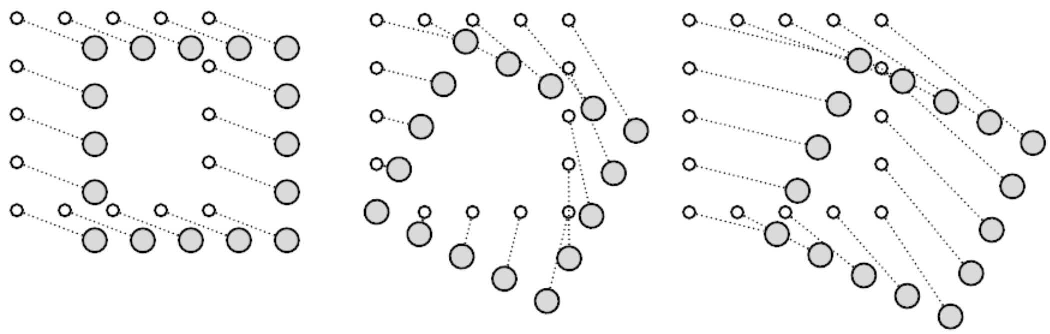

Figure 4.2.. A formation undergoing translation (left), rotation (middle), and simultaneous translation and rotation (right) [69]. Note that rotation requires varying path lengths, so velocities need to be adjusted between agents to keep the shape intact during movement.

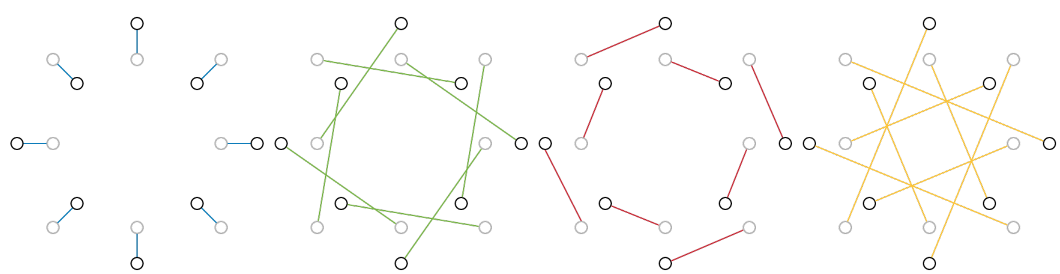

Figure 4.3.. Different ways move from one shape (grey) to another (black). The blue and green paths involve corner agents become side agents and vice versa, rounding out the shape while in motion. On the other hand, the yellow and red paths maintain positioning in the formation and maintain the shape during the transition. The green and yellow paths also feature more rotation, so the shape visibly shrinks in the first half of the motion and grow in the latter half.

An incredibly relevant example to this is drill design for marching bands, as they must plan for a variety of limitations while still maintaining cohesive formations. Transformations are limited by the speed at which a person can move while maintaining posture and formations must maintain minimum distances between people to prevent collisions. Transitions also depend heavily on extra frames between shapes to ensure the shape transitions cohesively or rely on those within the formation to understand what the intent is so that they can regulate their speeds and positioning while in motion [69]. Other entertainment-based applications that involve creating shapes with robots follow similar principles.

Self-Assembly and Structure Building

Self-Assembly is a unique type of formation in which agents physically attach themselves to each other to create a larger structure. This is a combination of both Patterns and Network Building as the formation rigid once formed while also adhering to a ruleset, in this case whether two agents have the correct interfaces and conditions to connect. Interfaces and logic are the most critical element for this type of task to ensure that agents know who to connect with and when. Loose emergent structures can rely solely on the ruleset, but automated self-assembly of a specific formation demands robust coordination algorithms to ensure correct agents are in the correct places and times. Should this larger structure need to move, principles from coordinated motion apply, but with much stricter requirements as spacing between agents are fixed. To build this construct, agents may need to first aggregate to an assembly location or assemble into subsections locally and then these subsections aggregate. Similar considerations apply in cases where the swarm construct needs to separate into sections, or completely decompose into individual agents.

Structure Building is the object focused analog to Self-Assembly where agents are tasked with placing objects in specific locations. In this case, the same principles apply where the goal is either a well-defined structure that requires coordination to place objects in the right place, or a ruleset generated structure in which objects are placed based on existing placement. Ruleset based structure building is often seen in the construction of ant colonies or insect nests where similar structures are consistently generated despite being unique. A caveat with ruleset-based construction is how sensitive it is to parameters, making tuning difficult by no less critical. As noted by Bonabeau, not all combinations of rules create a cohesive structure [70].

4.1.4 Search

Search is unique from the other three tasks in that it is focused on finding an object, location, or state that meets some criteria instead of the arrangement of individual agents and objects. *Aggregation*, *Dispersion*, and *Formation* tasks have a known goal and firmly established end points, but *Search* is based on a lack of sufficient prior information. If *Aggregation* and *Dispersion* form an axis based on agent positioning, *Search* would begin to cover an orthogonal axis where the goals are not known ahead of time.

Search is also unique as the idea of searching is not exclusive to swarms. In fact, search is often done with single monolithic systems. However, despite being something that can be done by a single agent, multiple agents provide new possibilities that single agents cannot attain. Simple concepts take advantage of numbers to reduce the time needed to find a target, but *Search* can also take advantage of nonlinearity and emergence in the swarm as well. Some of the multi-agent methods used to facilitate emergent searches are so effective that they are used as optimization algorithms today.

Based on the flexible definition, *Search* covers a wide variety of tasks and applications. Many of them feature similar components. Every case involves some target. This can be some agent or object, location, time, other state parameter, or even any combination of these. This target usually has some identifying features that make it distinct from other entities which must be detectable by agents tasked with *Search*. In some cases, these features are trails that can be followed or used to predict where a target is located. State based targets can be identified by moving along gradients for instance, while entity-based targets can leave trails or have associated signs that can be used to help locate it.

As an task orthogonal to the other three types, it can sometimes be an aspect of the other tasks. Object-based tasks in particular can have a *Search* element if object locations or states are not known ahead of time. Agents may also need to ability to search the swarm or formation they are in to find suitable locations to complete its task like with Network Building. Aggregation sometimes requires that agents first search for a suitable aggregation point.

To facilitate search, agents need to have some ability to classify information to separate useful information from environmental noise. Coordination is not necessary but can be highly useful to speed up the task and prevent agents from moving to previously checked locations. In general, the more information is known ahead of time, the faster the search ends, but this is not always possible, so the swarm needs the ability to modify its knowledge during missions to improve their search capability as they learn more. Common applications in literature that heavily involve *Search* are described below.

Blind Search

Blind Search is a task in which agents are tasked with finding an object, location, or state that meets some criteria with no prior knowledge. Without prior knowledge, this is essentially a *Dispersion* based Exploration and Mapping problem where agents attempt to map out an area, only stopping if a target is located. In cases where there are multiple known targets, this continues until all targets are found. If the number of targets is known, then the problem is identical to Exploration and Mapping, with the additional data of where targets are located. In cases where these targets are intermittent, then there are usually signs indicating where a target can potentially be, which need to be monitored or studied to make sure it fulfills all criteria for a target. This monitoring can be accomplished with multiple passes or leaving an agent or sensor package behind.

Source Search

Source search is a type of search similar to Blind Search, but with additional information in the environment. In the standard source search problem, there exists a source that is producing some change in the environment that can be detected and tracked. In cases where the trail is akin to a path, the method involves locating this path and then following it to find its source. This trail can potentially have false paths, so agents may need to split it there are forks in the path. Some paths may end up being false positives, requiring a new search. For a source that is constantly producing some environmental modifier that spreads, this effectively creates a gradient in the environment that can be tracked using the same principles as optimization, with the source acting as a maximum or minimum. This gradient based concept has even manifested as its own computational optimization algorithm, particle swarm optimization, which can be used to track multiple sources [71]. Should the environment be turbulent, gradient descent becomes less useful, so new methods are required, like looking for a trace of the target in the environment and then following the slope, wind direction, or some other environmental feature. Given the turbulence, it is possible that the trace is lost, require the search to start anew.

Detection

Detection is a bit of a reverse search in which agents actively monitor their surroundings until an entity or event meeting targeting criteria is detected. This can be accomplished using stationary agents placed in a specific or autonomously generated formation, or through the use of Patrolling agents. This can be achieved by having agents set up in a perimeter for cases where the target enters an area, or through fully coverage of an area for an event that can occur within the environment. This task is usually maintained indefinitely in the case of sensor networks whose sole purpose is to detect. In cases of moving targets, the agents can also collectively track the movement of this entity. In all cases, some form of sensor fusion is necessary to deal with signal noise and create as accurate of a track as possible.

Emergent Search

While all of the previous *Search* tasks can be accomplished by a single agent, Emergent Search is unique, requiring multiple agents and involving the identification of a nebulous concept instead of a well-defined target. Emergent Search is based on finding optimal paths to a target (on lop of location targets), which is a monumental task that a single agent can not effectively manage. With multiple agents, however, not only are there multiple agents for faster exploration, feedback loops can be used to generate solutions over time. This concept can be seen in foraging among ant colonies. While a single ant is capable of performing a search and locating a source of food, it has no concept of optimality or any reason to seek it. However, as it moves, it produces a pheromone to help find its way back to the colony. As a single ant moves along this path multiple times to forage for food, this pheromone trail gets stronger. If multiple ants forage however, the ant that makes the shortest trip is able to make more trips in the same amount of time as other ants, gradually creating a stronger trail over time through a positive feedback loop. Given enough ants, this can generate an optimal path automatically. This principle is so effective at creating paths it has also become a form of optimization, called ant colony optimization [14].

4.2 Objectives

Objectives are specific concepts and methods that influence how swarms perform tasks. While *Tasks* refer to specific actions taken by a swarm and swarm agents, *Objectives* focus on swarm metrics and performance, and by extension, swarm purpose. Many of these correspond to attributes or qualities commonly referred to as system "-ilities" (note, all of these do not end in "-ility") [72, 73]. Some of the more common ones are *Efficiency*, *Adaptability*, *Reliability*, *Usability*, *Flexibility*, *Resilience*, *Scalability*, and *Modularity*.

These are often qualitative and difficult to explicitly measure. Despite this, *Objectives* have a large impact on whether or not a swarm's performance is sufficient. These also emphasize different qualities and are often in conflict with each other. For example, *Efficiency* and *Resilience* are often incompatible, as one focuses on optimal use of swarm resources and leads to fragile systems, while the other utilizes redundancy which increases resource usage and costs. Both of these are desirable qualities, so often times balance needs to be struck between competing objectives in order to make a good swarm.

The next sections explore the *Objectives* listed above, including which categories they are most impacted by. Some swarm specific metrics are also discussed as well. Many more *Objectives* exist, but this study does not explore those in detail for the sake of brevity.

4.2.1 Efficiency

Efficiency describes how well swarm resources are used and is generally addressed through optimization. This can be achieved by minimizing the amount of resources used during operation or maximizing a particular output. As swarms often feature many types of resources like costs, agents, and time, as well as many types of output, efficiency may need these separate features in addition to other objectives. Optimizing for time may lead to increased costs for instance. As swarms need to operate within certain margins, some of these resources can be used as constraints in the optimization problem. In other cases, a balance needs to be struck, which requires multi-objective optimization methods [74]. Efficiency can be achieved through algorithms designed to improve use of a particular resource or metric, or by modifying characteristics at all levels of a swarm. Categories that directly address swarm behavior like *Dynamics* tend to have as strong impact on efficiency. Points of centrality identified in the *Network* characteristic group may also be useful to address efficiency. In most cases, this *Objective* runs counter to most others, which tend to demand more resources.

4.2.2 Adaptability

Adaptability describes how well a swarm reacts to unexpected changes and disruptions. From a characteristic level, this is primarily addressed through a swarm's existing contingency plans and ability to develop new ones, which are directly discussed in the *Intelligence* and *Dynamics* categories, but are also dependent upon a swarm's *Composition* to support it. This *Objective* is highly desirable for swarms that need to operate in conditions where prior knowledge is not achievable.

4.2.3 Reliability

Reliability addresses how likely a swarm will fail during normal operation and is usually affected by hardware and logic. Hardware based reliability is impacted by the limitations of agents and their hardware and how often these limits are met or exceeded. This also includes the *Network* structure of the swarm as well. Logic reliability on the other hand focuses on whether or not logic functions as intended and if there are any cases where the swarm has an error that prevents it from functioning. Many of these issues develop in particularly complex swarms where there are more places for errors to hide or for unexpected effects to occur.

4.2.4 Usability

Usability emphasizes the human element of swarms, focusing on ease of use. This can be how easy it is to deploy a swarm, collect agents after the mission, or operate the swarm as a controlling entity. These depend on the functions within agents and the swarm and whether they are designed with this in mind. From an external control perspective, centralized swarms tend to be easier to control due to there being a single point to influence agents, but this is limited by how many ranks exist on the hierarchy as more ranks lead to less control. For decentralized swarms, control is less direct and primarily set through interfaces or environmental features that influence swarm behavior. In cases where understanding a swarm is critical, this puts a limit on the allowable complexity of a swarm.

4.2.5 Flexibility

Flexibility addresses how many different things can a swarm do. This overlaps with Adaptability but focuses on planned changes instead of unexpected ones. Much of this is dependent upon swarm characteristics at all levels, and usually results in increased costs and complexity, which can be offset by the increased capability that comes with it. Increased flexibility also allows for more adaptable swarms as well, but too much focus on flexibility may lead to a swarm that is average at performing the tasks it was design to do. Like Adaptability, highly flexible swarms may be best used when exploring a concept or space in order to design more efficient, specialized swarms later.

4.2.6 Resilience

Resilience describes how likely a swarm is able to keep functioning after a significant disruption or even recover. This is mostly dependent upon *Network* characteristics, specifically *Hierarchy* and *Centrality*. Loss of higher ranked or central agents tends to disrupt a swarm more than loss of less connected or lower ranked agents. This can be addressed by having agents that can fill in for those roles should an important agent be lost but this increase in redundancy comes at higher cost. A similar effect can be observed from a *Composition* perspective where the lost agent is the only one of its type or one of a few, but this tends to overlap with *Centrality* as these agents tend to have relationships with many other agents. Some resilience is desired in order to take advantage of SoS concepts like stable intermediate forms when dealing with swarms that need to be constantly operating. In these cases, the swarm needs to be resilient enough that failing agents can be replaced by newer, possible better agents, without failing.

4.2.7 Scalability

Scalability impacts a swarm's ability to function across multiple scales and is directly discussed in the *Scalability* subcategory of *Composition*. In context of applications, it addresses the limits in which a swarm can continue functioning without a significant change in behavior. Increased scalability is desired to create general solutions that can be applied to more problems (and therefore reduce cost), but this can lead to suboptimal performance.

4.2.8 Modularity

Modularity addresses a swarm's ability to swap out characteristics, agents, or behaviors without needing to reconfigure everything. This is related to Resilience but focuses on planned changes instead of unexpected losses. Modularity is able to give swarms flexibility without requiring individual agents to support multiple conflicting capabilities, instead opting to swap in specialized agents as needed. Because components are separate, it is easier for swarms to swap agents out without interrupting the operation of the swarm, but this still requires well designed interfaces and the ability to function without some agents for a period of time. The use of consistent interfaces is critical to make sure that new swarm agents can be integrated without requiring drastic changes or retrofitting. A drawback to this could be the limitations of older interfaces, but transitions can still be done where new agents have both the old and new standards, with the eventual goal of phasing out the old interface.

4.3 Applications within Aerospace

Many types of swarms exist in aerospace today, but as the cost of small aerospace vehicles like UAVs and cubesats become cheaper, even more swarm concepts are being explored across the world. In this section, different types of swarms using aerospace vehicles are explored to illustrate how the Taxonomy of Swarm Applications can be used to decompose a particular application into multiple parts that can be studied separately and later combined. Applications in aerospace span both air and space, but those that operate solely in an atmosphere or in orbit have specific characteristics as a result of its environment, so these applications are divided into aeronautical and astronautical applications.

4.3.1 Aeronautical Applications

With the advent of lower cost UAVs like commercial quadcopters, there has been an explosion of interest in swarm applications. Many of these applications take advantage of the inherent mobility of aerial vehicles, using them as sensor platforms for *Search* and *Exploration* based tasks. Other applications use mobility to provide coverage across massive scales that would be impossible or too demanding for ground-based systems, making them fit for *Dispersion* tasks as well. On the other hand, the fragility of aircraft prevents them from congregating without risking catastrophic collision, making them poor agents for *Aggregation* tasks.

Many of the proposed applications can be grouped into categories by whether they are civilian or military applications. This is not a particularly meaningful way to separate the applications into groups with common methods like the *Task* breakdown, but does indicate if a swarm needs to consider combative or competitive elements. The applications that are explored in this section are listed below.

- Civilian

- Communication Network
- Aerial Mapping
- Weather Monitoring
- Disaster Search and Rescue
- Infrastructure Inspection
- Agricultural Applications
- Displays and Shows

- Military
  - Overwhelm
  - Distraction and Decoy
  - Surveillance
  - Strategic Formation
  - Loyal Wingman

It is important to note that this is by no mean comprehensive and misses a few common applications that are beyond the scope of this paper. One such application that is aerial delivery using a fleet of delivery vehicles. Though it possesses many elements that this framework can capture, the application involves a highly dynamic environment that includes active agents, making it a type of large-scale traffic management problem that would require considerations of characteristic levels higher than the *Group* level. The rest of this section gives a brief description of each application. A table that maps each of these to the previously described *Tasks* and *Objectives* is shown in figure 4.4.

Communication Network

Communication Networks utilizing swarms of aircraft have the advantage of being mobile, allowing coverage to shift as needed without limitations posed by terrain. On the other hand, they can potentially require more agents than ground-based networks due to the need to recharge. High winds and precipitation can potentially disrupt their function as well. Overall, they are best for highly dynamic and temporary coverage. This application primarily involves *Dispersion* and *Formation* tasks and involve *Cost Efficiency* and *Reliability* objectives.

			Tasks				Objectives									
			Aggregation	Dispersion	Formation	Search	Efficiency Cost	Efficiency Resources	Efficiency Time	Adaptability	Reliability	Usability	Flexibility	Reslience	Scalability	Modularity
Air	Civilian	Communication Network		x	x		x				x					
		Aerial Mapping		x	x	x	x		x		x		x	x		
		Weather Monitoring		x	x	x				x	x	x	x	x	x	x
		Disaster Search and Rescue		x	x	x			x	x	x	x	x	x	x	x
		Infrastructure Inspection				x	x				x	x		x	x	x
		Agricultural Applications		x	x	x	x				x	x	x	x	x	x
		Displays and Shows			x		x	x			x	x		x		
	Military	Overwhelm	x		x		x	x								
		Distraction/Decoy	x		x			x		x	x					x
		Surveillance		x	x	x	x				x			x	x	
		Strategic Formation		x	x				x	x	x			x		
		Wingman		x	x			x		x	x	x	x	x		x

Figure 4.4.. Mapping aeronautical examples to the different types of *Tasks* and *Objectives* described in the Swarm Framework

Aerial Mapping

Aerial mapping takes advantage of vehicle mobility to quickly map large swaths of an area. Changing vehicle altitude has the additional effect of changing the resolution of the image, allowing highly detailed imagery at a slower rate. Multiple angles also allow for 3D models to be generated from 2D images. This commonly involves combinations of different *Dispersion*, *Formation*, and *Search* tasks and various objectives depending on the specific context of the problem.

Weather Monitoring

Weather monitoring is similar to aerial mapping, but instead of physical images, this focuses on atmospheric data across an area. This information can be highly useful for monitoring and predicting severe weather like tornadoes, potentially expanding the warning time for dangerous events or providing critical information for manned aircraft in the air [75]. This primarily involves *Dispersion* and *Formation* depending on the kind of information needed, but can also include *Search* to locate particular phenomena. Given the turbulent environment swarms can be deployed in, this covers a wide variety of objectives as well.

Disaster Response and Search and Rescue

Disaster response and search and rescue typically involve situations where there is little infrastructure, or any existing infrastructure is damaged or destroyed, making aerial vehicles a powerful tool when seeking to explore and monitor and area. As temporary events, this is also less burdened by vehicle requirements like recharging. Mobility also allows for faster search and rescue missions unburdened by terrain. Different vehicles can be deployed for different requirements, making aerial swarms almost perfect for handling the variety of problems posed by disasters. This too involves *Dispersion*, *Formation*, and *Search*, as well as a great many objectives.

Infrastructure Inspection

Due to the sheer scale of some pieces of infrastructure or the inaccessibility of others, manual exploration with a human or a ground-based vehicle is incredibly slow or not possible. Some locations feature no platforms for vehicles to perch from, like nuclear plants, bridges, or wind turbine blades, requiring complex machines just to gain access to a location [76, 77]. Other types of infrastructure, like pipelines, span hundreds of miles, making it a costly and time-consuming endeavor. Both of these can be addressed using aerial vehicles which can easily maneuver into hard to reach locations and have to mobility to quickly scan large scale works [78]. In cases where aerial vehicles are unable to provide a detailed enough analysis, some researchers have even developed aerial vehicles that can shift into wheeled vehicles that can climb walls by using thrust to push against a surface [79, 80]. Given the mature of these applications, they are primarily focused on *Search* based tasks.

Agricultural Applications

Agricultural applications are similar to infrastructure applications in the sense of scale. Commercial farms typically feature thousands of acres of crops, making them difficult to manage efficiently. Large equipment can address high output tasks like planting seeds or watering, but applications like looking for diseased plants in a massive field are best suited to aircraft. Swarms of aircraft can also offer highly controlled precision for tasks like soil sampling and crop management, something that would be laborious for a person and impossible for heavy equipment [81]. These applications utilized all tasks except for *Aggregation*.

Displays and Shows

In addition to inspection, mapping, and mobile infrastructure, aircraft can also be used for purely artistic or entertainment-based applications. Swarms of aircraft can be used to create airborne images or displays, like the Intel drone swarm in the 2018 Olympics [82]. The ability to turn empty space into a canvass has great potential for the entertainment industry. Marching bands are already stables in college football games. With decreases in vehicle costs, aerial displays will likely become more popular. This is primarily focused on *Formation*, and involves objectives like *Efficiency*, *Reliability*, *Usability*, and *Resilience*

Overwhelm

Overwhelming is a military idea where some assaulting force has so many agents that either sensors or defenses are unable to keep up [4]. The simplest form of this involves sheer numbers alone, making it a type of *Aggregation* based task, but these groups can also be organized in specific ways to target blind spots or key sensors and defenses, giving some aspect of *Formation* as well. Given the combative nature of this task, many agents are lost by design, making *Cost Efficiency*, and *Resource Efficiency* key objectives, as well as other objectives like disposability.

Distraction and Decoy

Distraction and Decoy applications are a type of Overwhelming in which there is a decoy or distraction swarm meant to draw attention and focus of defense, while a separate group slips in unnoticed [4, 21]. As a result, it has similar considerations to Overwhelm, but also considers *Adaptability* and *Reliability* as well. For cases where there are some decoy agents can shift roles once an opening is found, some level of *Modularity* may be desired as well.

Surveillance

Surveillance in military contexts are a defensive tool in two ways. Active surveillance allows a system to local and track threats, giving the system a chance to respond as the situation demands, exhibiting the *Search* task Detection. The surveillance network also provides a psychological defense in the form of a deterrent, keeping threats out through the fear of being attacked [4]. Overall, this takes advantage of *Search* tasks as described as well as some form of *Dispersion* and *Formation* to optimize sensor placement, and values *Reliability* and *Resilience* above all.

Strategic Formation

Strategic formations are a tactical use of aerial vehicles in the vein of soldier formations. Arranging agents into specific shapes can provide benefits like hiding the true size of the swarm or making the swarm less vulnerable in certain angles. Larger formations like waves can also be used to simply management while also maintaining power, and having multiple formations can allow for targeted strikes in multiple locations [4, 8]. Such targeted strikes have been successful in the past with soldiers and have the potential to be more dangerous, especially when the agents themselves are much more disposable.

Loyal Wingman

Aerial swarms also have potential use in loyal wingman applications where a single manned fighter is accompanied by a group of semiautonomous aerial vehicles that either automatically support or follow instructions issued by the manned fighter [83]. This concept involves many of the ideas outlined in the framework, but many also require some characteristics from higher levels to factor in the human element for particularly complex systems. Otherwise, the manned fighter can be treated as a commander within the swarm. By nature, this makes the Loyal Wingman concept a highly coordinated *Formation* based idea, but can involve some level of *Dispersion* in the distributed function context as wingmen typically rely on a commander. Due to the many objectives of a fighter, multiple objectives are involved as well.

4.3.2 Astronautical Applications

While aircraft swarms have only just become possible, spacecraft swarms have existed for some time through satellite constellations. These swarms have been able to provide various services across the planet for decades and are a firmly established piece of infrastructure. As satellite costs drop however, more companies seek to launch their own constellations or highly localized satellite clusters into orbit for their own purposes. Lower costs have also opened space exploration up to swarm architectures designed to explore and study other celestial bodies at greater precision than before.

Orbital swarms feature heavily restricted *Locomotion* characteristics and are largely defined by the orbits they operate in and station keeping methods to maintain this. Their vantage point makes them well suited to a sensors applications, necessitating good *Sensor* and *Transmission* characteristics. For sufficiently large constellations, some autonomy may be required to reduce management costs. Specifics largely depend on the orbits and formations that vehicles are designed for. Lower altitudes may have a shorter lifespan due to atmospheric drag, while higher orbits may require better sensors. Satellite formations like constellations, trains, and clusters all have different uses as well [84]. Exploration swarms largely depend on the agents. Orbital based exploration follow similar principles to other orbital swarms, while terrestrial, aquatic, or aerial swarms feature their own unique requirements.

For this study, astronautical applications are grouped into constellations, clusters, and exploration swarms. There is no significant reason for this categorization other than scale and purpose of the swarm. Constellations span a planetary body and are generally meant to act as infrastructure intended to last for long periods of time with new agents being launched to replace failing ones. Clusters are more localized and have no such requirements. Exploration is unique in that they are meant to be launched long distances with a greater ability to change orbits as needed during missions. Specific applications within these groups are listed below.

- Constellation
  - Sensor Constellation
  - Communication Constellation
  - GPS Constellation
  - Power Transmission
- Cluster
  - Fractionated Satellite
  - Orbital Debris
  - Synthetic Aperture
  - Space Construction
- Exploration
  - Terrestrial Exploration
  - Site Preparation
  - Orbital Exploration
  - Asteroid Mining

Like with the aeronautical applications, this is not an exhaustive list. The rest of this section gives a brief overview of some of the existing or more commonly proposed swarm applications within space. A table that maps each of these to the previously described *Tasks* and *Objectives* is shown in figure 4.5.

			Tasks				Objectives									
			Aggregation	Dispersion	Formation	Search	Efficiency Cost	Efficiency Resources	Efficiency Time	Adaptability	Reliability	Usability	Flexibility	Reslience	Scalability	Modularity
Space	Constellations	Sensor Constellation		x	x	x	x	x			x			x	x	
		GPS Constellation		x	x		x	x			x	x		x	x	
		Communication Constellation		x	x		x	x			x			x	x	
		Power Transmission		x	x		x	x			x	x		x	x	
	Clusters	Fractionated Satellite		x	x			x			x		x	x		x
		Synthetic Aperture		x	x	x		x			x				x	
		Orbital Debris	x		x	x	x	x		x	x		x	x	x	x
		Space Construction	x		x	x		x		x	x	x	x	x	x	x
	Exploration	Terrestrial Exploration		x	x	x	x	x	x	x	x		x	x		
		Orbital Exploration		x	x	x	x	x	x	x	x		x	x		
		Site Preparation	x		x	x		x		x	x		x	x	x	
		Asteroid Mining				x		x		x	x		x	x	x	

Figure 4.5.. Mapping astronautical examples to the different types of *Tasks* and *Objectives* described in the Swarm Framework

Sensor Constellation

Sensor constellations are swarms of satellites crossing the planet meant to record information from a high vantage point in space and transmit information back to Earth. These can be for civilian purposes like creating global maps, monitoring weather, or tracking animal migrations, as well as military purposes like spy satellites and other forms of surveillance. These typically involve evenly spaced orbits at identical inclinations for full planetary coverage at all times. As a constellation, it is primarily based on *Dispersion* and *Formation* tasks, but as a sensor platform, can also perform *Search* tasks if agents are equipped with sufficient attitude control. Along with most constellations, it also tends to focus on *Cost Efficiency*, *Resource Efficiency*, *Reliability*, *Resilience*, and *Scalability*.

GPS Constellation

Constellations of navigation satellites are responsible for the global positioning system (GPS), a critical piece of global infrastructure that can be used to give accurate location information. GPS Constellations are also highly critical components of many swarm-based systems used in other areas where positioning is critical like agriculture. Each of these satellites need to be able to locate themselves accurately and provide a signal that other devices can use to locate themselves on the planet. *Dispersion* and *Formation* are critical aspects of these swarms and tend to have the same objectives as sensor constellations in addition to *Usability*.

Communication Constellation

Communication constellations are swarms of satellites that act as communication relays, receiving signals and immediately transmitting them to their targets. They are similar to sensor and GPS constellations. As a result their *Tasks* and *Objectives* are almost identical to GPS constellations as well, with the only major differences being larger antennas and other communication hardware at the *Individual* level.

Power Transmission

Power transmission constellations are a hypothetical swarm of satellites that generate power from solar radiation and beam it back to the surface. This can also be done in clusters or trains, but as a piece of infrastructure, it will likely become a constellation over time. They can be placed around the Earth, or potentially around the Sun to form a Dyson swarm [85]. These constellations likely feature similar *Tasks* and *Objectives* as other constellations, with some focus on *Usability* as well.

Fractionated Satellite

Fractionated satellites, also known as federated satellites, are clusters of satellites moving along parallel orbits that each all work together to perform some function. Each satellite is specialized for some purpose and requires some of the others in the cluster in order to fully function. These are meant to reduce the cost of individual satellites or reduce launch mass and can collectively perform specific tasks. Depending on the orbit, some are fixed over a single point on Earth like geostationary orbits or meant to focus on a particular latitude like Molniya orbits. The potential uses of these are numerous, but involve some form of *Dispersion* and *Formation*, and possibly *Search*. They are meant typically designed for *Resource Efficiency*, and feature *Modularity* and *Flexibility*.

Synthetic Aperture

Synthetic apertures are created when several separate telescopes or sensors in different locations synchronize their phase data to produce images that would be producible by a much larger lens. These are highly useful when telescopes are unable to support a lens of a certain diameter, such as in space, or for particularly long-distance measurements like the Event Horizon Telescope. Orbital based synthetic apertures can potentially be used to simulate much larger lenses from space, reducing the overall mass required to send future telescopes into space. These require similar considerations to those of Fractionated Satellites, but has potential for *Scalability* as well.

Orbital Debris

A potential use of satellite clusters is management of orbital debris. A swarm of satellites can potentially locate and deorbit large pieces of debris through coordinated thrusting before separating and shifting back into a patrolling orbit to locate another piece of debris. As separate units, they can connect to a larger variety of surfaces and are much cheaper to replace. This is also one of the few aerospace use cases that can potentially feature *Aggregation* based tasks as it relies on pooling the thrust of multiple agents.

Space Construction

Space construction is a more complex variation of orbital debris management as the goal is not solely deorbiting objects but also maneuvering components and assembling them to create large structures. Detachable miniature space tugs can attach to construction materials in multiple directions, allowing objects to be carefully moved during construction. This features a combination of *Aggregation*, *Formation*, and *Search* to fully automate a structure building scenario, and requires consideration of many types of objectives like *Modularity*, *Usability*, and *Reliability*.

Terrestrial Exploration

Terrestrial exploration of another planetary body would functionally be no different that exploration of Earth aside from limitations like lack of an atmosphere (no aircraft) or lower gravity. With such a large design space for these applications, the particular needs depend on the mission as asteroid exploration is different from exploring the Moon, Mars, or Titan. Efficient search is a necessity due to the harsh environments and applications require *Dispersion*, *Formation*, and *Search* tasks along with a variety of objectives.

Orbital Exploration

Orbital exploration is similar to terrestrial exploration, but with the movement limitations imposed from being in orbit. As of now, these come in two types, flybys and orbiters. Flybys are part of a larger mission in which the satellites use the body of interest primarily for a gravity assist, which give a short window for satellites to make observations before continuing, like the Voyager flybys of the gas giants or the recent New Horizons flyby of Pluto. These typically offer little flexibility, making swarms a poor choice for these specific instances. If the body is the target however, they can decelerate into orbit around the body to perform long term observations like Cassini. Satellites in these applications typically come with a decent amount of fuel to give them to ability to orbit around several bodies, and can potentially be done by swarms, allowing for simultaneous measurements. As an exploration platform, these involve *Dispersion*, *Formation*, and *Search* and a variety of objectives.

Site Preparation

Similar to Space Construction, swarms of ground vehicles can be used to prepare a site in advance of manned missions. Autonomous vehicles programmed to clear a site of debris or start excavating space for habitation can save significant amounts of time needed to begin a colony. Specifics of this a still far in the future, but researchers have already attempted to explore the problem [86]. Such tasks involve *Aggregation*, *Formation*, and *Search* and multiple objectives, particularly *Reliability*, *Resilience*, and *Scalability* to manage large scale tasks in a hostile environment.

Asteroid Mining

Swarms can also be used to excavate and mine asteroids using similar ideas to site preparation. Depending on the resource (helium vs metals), swarms can be used in lieu of difficult to launch heavy equipment. The swarm itself may be too limited for deeper mining, but multiple vehicles can mine the surface of asteroids before moving to another nearby asteroid should one be close enough. This relies on *Search* as positioning is not as critical to the application aside from collision avoidance.

—

The Taxonomy of Swarm Applications is a powerful tool in this Swarm Analysis Framework that can be used to study the applications of swarms. Complex applications can be divided into different components that can be analyzed separately to gain a better understanding of how a particular swarm behaves, before tuning swarm characteristics identified using the Taxonomy of Swarm Characteristics to optimize performance. Whereas previous methods at understanding applications attempted to create packages of methods and methods tailored for a specific application, this gives a much more flexible understanding of what a swarm can do, allowing for complex tasks to be designed in a manageable way.

Whereas the Taxonomy of Swarm Characteristics outlines what a particular swarm is and what it can do, this taxonomy outlines how it does a particular application and in what ways this can be done to improve some performance feature. Collectively, they can be used to give a comprehensive overview of a swarm, its capabilities, and its priorities, establishing the conceptual design of a swarm.

5. CASE STUDY

To illustrate how the Swarm Analysis Framework is useful, a swarm-based mapping problem is explored as a case study. This chapter begins by using the Framework to analyze the problem and possible swarm characteristics, before modifying these to suit the testing environment used to study candidate swarm designs. Afterwards, the swarm is evaluated in this environment with a parameter sweep to observe behavioral trends as certain parameters are varied. Next, an informal analysis is done by varying different parameters to observe their impacts in swarm behavior. This gives an initial overview on how a particular group of swarm architectures behave when solving this problem, which can be used to guide further development.

Though the case study studies swarm performance, this is not meant to solve an actual mapping problem. The goal is to observe how variations in swarm parameters can greatly affect swarm behavior, so many simplifications are made. Results from this case study are not meant to make any definitive statements on the behavior of any particular swarm, and many characteristics that impact a swarm are ignored to reduce the complexity of the problem. Likewise, the methodology used in this case study is one of many different approaches that can be used when understanding swarms and is by no means the best way to solve similar problems. Though the Framework can be a highly useful approach when designing swarms, it does not address any design methodology within it as of this study.

5.1 Using the Framework

This section shows how the Swarm Analysis Framework can be used to understand how a swarm can explore and locate survivors in the aftermath of a natural disaster. First, the Taxonomy of Swarm Applications are used to break down and understand the problem at hand. Afterwards, the Taxonomy of Swarm Characteristics is used to systematically determine swarm characteristics that can be useful for this application.

The scenario is based on a hypothetical disaster response situation in the aftermath of Hurricane Katrina in New Orleans. Extensive flooding following Hurricane Katrina left much of the existing infrastructure damaged and roads impassable to land-based vehicles. In addition, hundreds of survivors were stranded in these flooded areas and required first responders [87]. In such a situation, one of the most important things to establish is an idea of where these survivors are and how they can be reached, necessitating some mapping capability. At the time, this was limited to helicopters, which simply did not have the numbers to provide the necessary coverage needed to identify survivors and help organize rescue efforts. With recent developments in aerial vehicles and AI/ML however, new tools exist that can be used to address this problem should it ever happen again. The environment is based off of the flooding data compiled by Louisiana State University after the disaster, as shown in figure 5.1 [88].

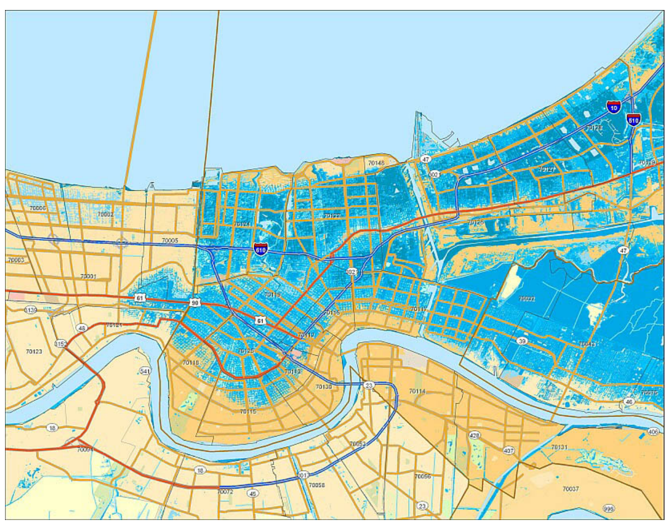

Figure 5.1.. Extent of flooding in New Orleans from Hurricane Katrina [88].

5.1.1 Application Analysis

The specific problem to be addressed is the mapping of the disaster zone. Using the Taxonomy of Swarm Applications, it can be shown that the problem is primarily a *Dispersion* task, specifically a exploration and mapping variant of it, which includes a basic element of *Search* (identifying unexplored areas). As a result, any swarm needs to have methods to avoid each other to reduce overlapping coverage and identify places that have not been explored. Although New Orleans had previously been mapped, the flooding damage is extensive enough that the *Search* aspect is similar to a *Blind Search* as many former landmarks are likely unrecognizable. Depending on if refugees are mobile and able to provide signals, aspects of *Source Search* and *Detection* may be involved as well, combined with the *Dispersion* task *Patrolling*.

From an *Objectives* standpoint, because there are human survivors involved, *Time Efficiency* is likely a key objective to make sure survivors are identified as soon as possible. Due to the large disaster zone, some level of *Scalability* is probably useful as well. Due to lack of surviving infrastructure, *Reliability* is also key in order to operate. Though other objectives can be relevant in a general disaster response, they are not as immediately relevant for the mapping aspect of it.

5.1.2 Characteristics Analysis

Development of a swarm design can be done from a top-down perspective beginning with swarm characteristics and behaviors and choosing agents that fit within these, or from a bottom up perspective where agents are chosen first and swarm behavior is designed around these agents. This case study will primarily use a bottom up approach where agents and relationships are established, and swarm characteristics emerge from these characteristics, instead of being explicitly planned. However, some *Group* level characteristics, like *Homogeneity* and *Hierarchy*, are determined beforehand in order to establish an initial swarm architecture to work from, reduces the design space and giving a good starting point for developing agents and relationships. Other *Group* level characteristics on the other hand are generated and observed instead of designed. Due to lack of readily available data, heuristics and aggregate behaviors that would normally be used as guiding ideas are unavailable, so many of these swarm characteristics require testing to discover and understand.

Initial Group Characteristics

Out of simplicity, the swarm used in this case study consists of a number of identical vehicles, making this a homogeneous swarm. The number of vehicles is a useful variable to test in order to judge performance and *Scaling* behavior. *Capability* characteristics like *Access* match those of an individual agent, while *Range* depends on the *Physical* characteristics of swarm agents agent. *Output* characteristics require testing to establish aside from some predictions like maximum range, which is dependent upon vehicle speed and battery capacity.

From a *Network* standpoint, the agents feature no hierarchy aside from agent precedence based on the order of agents within the swarm. This only affects tasking (assuming there is coordinated tasking) in the sense that the first swarm agent has precedence when choosing locations, and each successive agent defers to previous agents. This is not optimal but reduces simulation times. Other *Network* characteristics like *Connectivity*, *Distribution*, and *Centrality* are not integral aspects of the behavior of this swarm, so these are not studied. Other characteristics like *Intelligence* and *Dynamics* are explored after the agents themselves are established.

Characteristic Analysis

Looking at the environment posed by the problem, it can be inferred that terrestrial and orbital agents are ineffective. Aquatic agents can potentially be useful to traverse the flooded areas but have a limited line of site and are restricted to flooded areas. Aerial agents are probably the best option to explore the disaster zone due to their high mobility and indifference to terrain. Since careful observation is desired, some hovering capability is likely necessary as well, meaning that the best agents are likely rotorcraft instead of fixed wing vehicles that require motion to stay in the air. This immediately establishes basic *Locomotion* and *Movement* characteristics for swarm agents.

As an exploration problem, agents require visual sensors like cameras to map areas, establishing baseline *Sensing* characteristics and setting limitations on agent altitudes. Some basic *Transmission* is needed to establish controls and live feeds to ground stations as well, and *Memory* is a requirement to store mapping data as backup. It can be assumed that most planning is done prior to deployment or managed by an external entity so individual *Processing* is probably not highly critical aside from enough to support basic autopilot and wind resistance functions. *Manipulation* is likely unnecessary aside from a way to adjust cameras and sensors. Due to the scale of the problem, *Power* is a constraining factor so higher battery life is preferred. Aircraft structure needs to be light in order to improve vehicle speed, but other features are probably not important aside from some static stability characteristics.

For *Functional* characteristics, *Information Exchange* is simple, with agents actively transmitting information a central location at all times, while also storing it in memory cards for backup. *Control* is likely through an autopilot system based on GPS waypoints or other method for path control. *Control* beyond this, like altitude changes, likely originates from a central authority. *Disruption* logic like wind resistance is necessary to keep an agent airborne and in position, as well as stabilizing logic for manually piloting. *Tasking* is determined either by a central authority or by agents. Internal agent-based tasking can be done by comparing previously explored areas, unexplored areas, and distances to them. In terms of *Self-Awareness*, agents only really know of their position and velocity, and *BDI* is simple, based on an agent's need to explore and recharge. These can be seen in figures 5.1, and 5.2.

Because *Network* characteristics are not critical, *Interaction* characteristics are probably sparse. All agents need to possess some level of *Swarm Awareness* in order to prevent collisions, but this can be accomplished from keeping track of the positions

Table 5.1.. Physical Characteristic analysis for a modified DJI Mavic Air 2 quadcopter used in the Disaster Response Case Study [43].

Subgroup	**Characteristics**
Locomotion	Quadcopter mobility; aerial vehicle with limited wind resistance; top speed of 19 m/s
Manipulation	3-axis gimbal system for camera
Sensing	4K resolution camera; forward, backward, and downward facing cameras for piloting; RF receiver
Transmission	1080@p30fps live video feed
Processing	Enough processing power to support autopilot and stabilizing software
Memory	MicroSD card up to 256 GB
Power	LiPo 3s 3500 mAh battery; max flight time of 34 minutes
Structure	Plastic/metal/composite construction; folding frame; 183x253x77 mm dimensions; weighs 570 g

Table 5.2.. Functional Characteristic analysis for a modified DJI Mavic Air 2 quadcopter used in the Disaster Response Case Study [43].

Subgroup	**Characteristics**
Movement	Hovering and point turns capable, moves in loops within map tiles
Information Exchange	Always active controller signal
Control	Autopilot based on algorithms
Disruption	Not explored
Tasking	Agents choose nearest unexplored tile
BDI	Search for unexplored tiles, avoid collisions
Self-Awareness	None

Table 5.3.. Awareness Characteristic analysis for the Disaster Response Case Study.

Subgroup	**Characteristics**
Swarm Awareness	All agents are aware of each other
Environmental Awareness	Agents are aware of their terrain and can identify refugees
Prediction	Agents are not able to model each other's behavior

and velocities of other agents within range of sensors. *Environmental Awareness* is necessary to keep track of locations that have already been explored, but aside from that, knowledge of terrain is not critical barring significant gusts that force an agent off course. *Prediction* is unnecessary as long as avoidance and tasking algorithms tasking work. *Communication* characteristics like *Range, Bandwidth, and Latency* are important considerations if central authorities are located at map edges, and *Transmission and Receiving* for agent position and velocities is largely done through local broadcasts. Communication to central authorities may need some form of directed communication to overcome vast distances, however. *Role* characteristics are simple due to the lack of hierarchy and homogeneity of the swarm. *Purpose* and *Rank* are identical for all agents aside from the aforementioned tasking precedence, while *Cooperation* varies based on the swarm's chosen coordination level. These can be seen in figures 5.3, 5.4, and 5.5.

In terms of undefined *Group* level characteristics, most *Dynamics* characteristics and *Scaling* is difficult to predict ahead of time. Fortunately, *Latency* of both kinds are non-issues due to the lack of synchronization required. Without prior knowledge, *Evolution* and *Scaling* require testing to identify. *Capability* characteristics require a more defined swarm to determine, and 5.6, 5.7, 5.8, 5.9, and 5.10.

Table 5.4.. Communication Characteristic analysis for the Disaster Response Case Study.

Subgroup	Characteristics
Range, Bandwidth, Latency	Not limitation, no delays
Transmission and Receiving	UAVs communicate to those nearby to avoid collisions and to relief sites for coordination and tasking.

Table 5.5.. Role Characteristic analysis for the Disaster Response Case Study.

Subgroup	Characteristics
Purpose	UAVs seek to map unexplored areas and locate refugees
Rank	Vehicles possess no inherent rank aside from their number in the swarm, which denotes precedence in tasking
Cooperation	Varied within the study from no coordination to simple retasking algorithms; knowledge of explored locations assumed to be global

Table 5.6.. Composition Characteristic analysis for the Disaster Response Case Study.

Subgroup	Characteristics
Size	Variable number to be tested
Homogeneity	Completely homogeneous swarm
Scaling	To be studied

Table 5.7.. Capability Characteristic analysis for the Disaster Response Case Study.

Subgroup	Characteristics
Access	Vehicles can fly in unobstructed airspace and can see into any places visible from the air
Range	Limited by proximity to disaster relief sites for recharging
Output	Equivalent to range

Table 5.8.. Network Characteristic analysis for the Disaster Response Case Study.

Subgroup	Characteristics
Connectivity	Not studied
Distribution	Not studied
Centrality	Swarm network has no consistent centrality that matters, but disaster relief sites are central to swarm
Structure	No true structure that is relevant
Hierarchy	Agents at same level aside from tasking priority
Physical Networks	To be studied
Organization	No planned groups

Table 5.9.. Intelligence Characteristic analysis for the Disaster Response Case Study.

Subgroup	**Characteristics**
Consensus	Not studied
Coordination	Varied to evaluate effect on performance
Group Computation	None
Learning	None

Table 5.10.. Dynamics Characteristic analysis for the Disaster Response Case Study.

Subgroup	**Characteristics**
Information Latency	Ignored; information is assumed to be global available at all times for this study
Physical Latency	Not studied
Stability	Potentially exists, but is not a key focus
Evolution	Unknown, must be simulated

5.2 Testing Methodology

Using this Swarm Analysis Framework, this mapping problem can be divided into simpler tasks that can be further explored to identify useful methods for solving them, alongside key metrics like timing and scalability. In addition, based on the context of the problem, many agent and swarm level characteristics are systematically identified after making some architectural decisions, showing potential options for a swarm. With these characteristics established, an actual swarm can be developed and tested.

Before further design can begin, testing goals need to be established to identify what specific characteristics need further development. Due to the study;s scope, physical testing is impossible, so swarm architectures need to be simulated virtually. Unfortunately, one of the biggest challenges for studying swarms is the inherent difficulty in modeling them. Most analytic or empirical methods only capture individual or aggregate behavior, but rarely both and almost never the specific details. This is due almost entirely to the sheer number of interactions that occur within a swarm and resulting uncertainties. As described by Clough, swarms are often nonlinear, non-deterministic, and require observation [4]. Fortunately, a common SoS technique, agent-based modeling (ABM) is almost perfectly tailored for this problem.

Agent based modeling is able to capture individual and swarm behavior from the bottom-up by simulating agents in an environment and letting them exist and interact, effectively creating a virtual swarm that parallels the behavior of a physical swarm [89, 90]. Fidelity and complexity is also able to be controlled using ABM, allowing specific phenomena to be isolated and studied quickly, which is difficult using physical tests, which often takes significant time and resources to develop. Granted, the method does have drawbacks. A single simulation is incapable of providing trends so many trials are needed to find average behaviors at a specific design point, and multiple design points are required to find trends. There is also a limit based on computation time which grows exponentially as detail increases. Nonetheless, ABM can be a useful tool for generating the data needed to understand swarm behaviors and potentially develop heuristics that can be used later.

Given the limitations of ABM as a testing methodology, simplifications need to be made to the swarm. This is useful for a number of reasons. Reduced complexity lowers simulation time, which is critical when performing detailed studies across multiple parameters. Simplifications also allow observation of specific trends isolated from the myriad of other factors that can prevent subtle trends from being noticed. Though this is not accurate to a real-world test case, this isolation may be critical to understand key trends in swarm behavior that would be impossible to observe otherwise. Simplifications are also useful for brevity as the goal of this case study is not to demonstrate ways to observe swarm behavior.

The first simplification affects timescale. Instead of operating at a small timescale to fully model every decision made by agents, the simulation operates at five-minute intervals. By doing this, complex movement patterns to explore an area can be reduced to areas covered within five minutes, which is much simpler to calculate. Though these small interactions are important, the study operates under the assumption that these short timescale behaviors are dominated by aggregate behaviors.

Similarly, the area to be mapped is simplified. Instead of a continuous space for agents to explore, the map is discretized into an array of square kilometer tiles representing natural bodies of water, flooded land, dry land, and disaster relief sites based off of data recorded in the aftermath of the Hurricane [88]. This can be used in conjunction with the simulation timestep to turn a complex local exploration problem into a simple "number of tiles explored per timestep" problem. The overall problem also becomes "how long does it take to explore every tile." This simplified map can be seen in figure 5.2.

With the testing environment established, simplifications can be made to the actual application and swarm characteristics. Since this is an initial study, it would be best to simplify the complexity of the application. As a result, the case study assumes that any refugees in the disaster zone are stationary and immediately visible

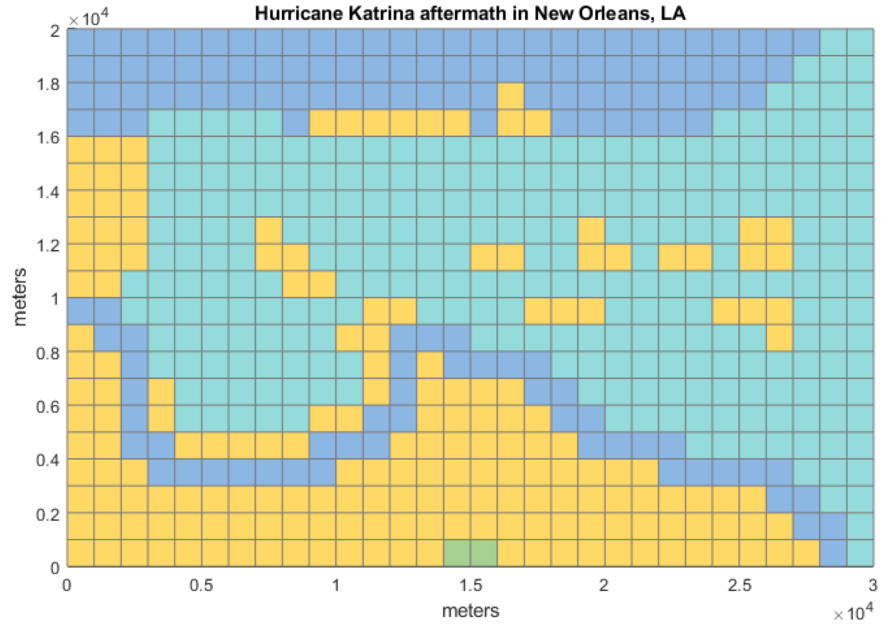

Figure 5.2.. Discretized map of New Orleans post Hurricane Katrina. Blue corresponds to natural bodies of water, teal to flooded land, yellow to dry land, and green for disaster relief sites. This is based off of actual flooding data [88].

when an agent is in range. The priority is also to explore the map as quickly as possible instead of actively seeking refugees. This brings the many potential tasks down to one: a *Blind Search*. The primary metric to be studied is *Time Efficiency*, with some interest in *Scalability* to better study how swarm behavior changes as parameters vary. Agents are assumed to be perfectly reliable, with no disruptions from wind or uncertainty in measurements.

From a characteristic standpoint, all movement characteristics are simplified as a point-to-point average line of motion between timesteps. Since agents do not have to worry about terrain, this should not be too much of a stretch. Sensors are assumed to be enough to map out an entire tile within a timestep, and communication parameters at both the *Individual* and *Interaction* levels are assumed to be enough to support any ranges and bandwidth required. *Power* and *Memory* remain unchanged. *Intelligence* characteristics like *Coordination* and *Consensus* directly addressed in tasking algorithms, while *Learning* is not explored.

In a more realistic instance, *Communication* is largely dependent upon the ranges available to sensors and constitutes a significant portion of the swarm logic in real time. Agents that notice points of interest can communicate to others to request their presence to get more information for instance. However, this is not captured very well using the grid-based map, as the actual timing of the simulation will be similarly discretized. *Awareness* also needs to be considered more for a physical system to prevent collisions. This can be done through external coordination if agents lack the ability to locate each other.

Based on this, *Locomotion*, *Movement*, *Sensing*, and *Tasking* characteristics are the most important for this particular study, with *Power* and *Memory* acting as constraints to each agent. With amount of data that can be stored on memory cards today, it can be assumed that power is more restrictive overall, allowing *Memory* to be disregarded as well. In later iterations of this problem, the ignored characteristics can be factored in to improve fidelity.

5.2.1 Swarm Design

With these simplifications made, specific agents can now be designed or selected from existing vehicles. Designing agents specifically for this problem is ideal, but this requires development time and extensive testing, which is expensive. Selected from off-the-shelf vehicles on the other hand are much less costly but agents will likely be suboptimal for the application. A potential compromise is to take existing systems and retrofit them, which is used for this study. The agents used for this case study are based on DJI Mavic 2 quadcopters, theoretically modified to include increased processing power and autonomous behavior. *Physical* and *Functional* characteristic breakdowns are based on figures 3.1 and 3.2 on page 39. These characteristics are used to create virtual agent models in MATLAB for simulation.

Based on the already established *Group* and *Interaction* characteristics, the overall architecture of the swarm can be seen in figure 5.3. However, as the vehicles need some way to recharge, this swarm is not enough. Supporting infrastructure is also needed to keep the swarm operating. In this case, this matches the locations of the disaster relief sites that may be set up in the aftermath. In terms of analysis, these can also be factored into the swarm as a set of stationary agents, but because these are not explicitly within the control of the designer, they are excluded from the formal swarm being studied. During operation, each agent keeps track of the closest site available to it, creating a dependence that varies over the duration of the application. The complete architecture can be seen in figure 5.4.

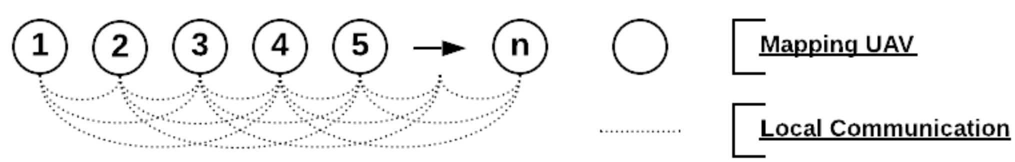

Figure 5.3.. Model of the Mapping Swarm in the case study.

Based on the current swarm architecture, two parameters are left as testing variables: the number of vehicles (*Composition*), coordination between agents(*Intelligence*). The number of vehicles is simple to represent, but coordination is represented by how many times an agent chooses a new destination in the case its original is “claimed” by an agent with tasking priority. No retasking allows agents to choose the same tile, while retasking should theoretically prevent this. Returning to the objectives outlined, the time to explore the map is the main priority and is the main metric for the various swarm parameters being tested. *Scaling* is observed in the behavior.

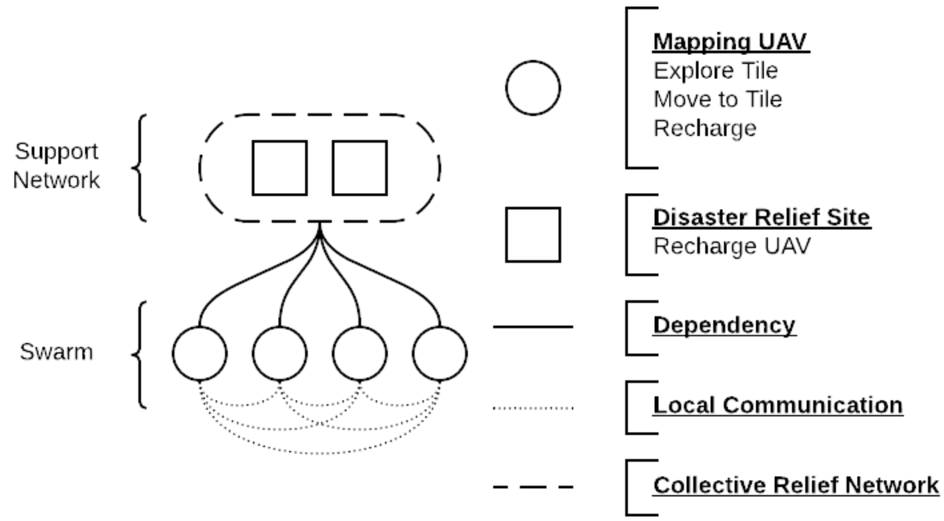

Figure 5.4.. Model of the Mapping Swarm and how it connects to the support network of Disaster Relief Sites for recharging.

In terms of agent behavior (*Tasking*), agents follow a cycle. First, an agent locates the nearest unexplored tiles to its current position, calculate how much power need to travel to each tile, explore it, and return to a disaster response site to recharge. Based on whether this value is less than the agent's current power remaining, it randomly chooses one of these as a destination. If the swarm allows for retasking, an agent checks to see if its chosen destination is identical to that of another agent. In the case that it is, the agent restarts the destination search process but ignore the claimed tile in the process. An agent does this as often as coordination allows, anywhere from one retasking cycle at maximum, or no limit. Afterwards, the agent travels to the destination tile and begin moving around the tile to explore it and locate survivors. One the tile is explored, it begins the destination search process again. If no locations are within range, it returns to the nearest response site to recharge.

1. Search for candidate destinations.

2. Calculate power requirements to go to these destinations.

3. Compare power requirements to power available.

    - If power requirements exceed power available, remove destination.
    - If power requirements exceed power available for all locations, set destination to nearest recharge site.
    - If power requirements are less than power available, randomly choose from the three closest destinations.

4. If retasking limit is not reached, check if destination has already been claimed by another agent.

    - If chosen destination is claimed by another agent and the maximum number of retaskings has not been met, move back to Step 1 but ignore the previously chosen destination. Raise the retasking counter by one.
    - If the destination has not been claimed by another or retasking counter is at the maximum value, proceed as normal.

5. Begin moving to destination.

6. Arrive at destination.

    - If destination is an unexplored tile, move to explore the tile.
    - If destination is a recharge station, land begin recharging or swap out battery for a charged one.

7. Repeat until map is fully explored, no locations are in range, or all agents fail.

5.3 Simulation and Results

The simulation is based on the map in figure 5.2, with all agents beginning between the two green disaster relief stations at the south end of the map. From here, agents move from tile to tile until the map is fully explored, recharging when needed. Distance traversed on the way to a destination is based on the velocity of the agent and the timestep. Graphically, the map starts greyed out, but changes to a more saturated color as individual tiles are explored. Vehicle paths are traced in grey to show how the swarm behaves as it explores. Some tiles also have refugees, which is revealed as agents explore a tile. These refugees are assumed to be visible, constant in number, and stationary. Snapshots from an example run can be seen in figures 5.5, 5.6, and 5.7.

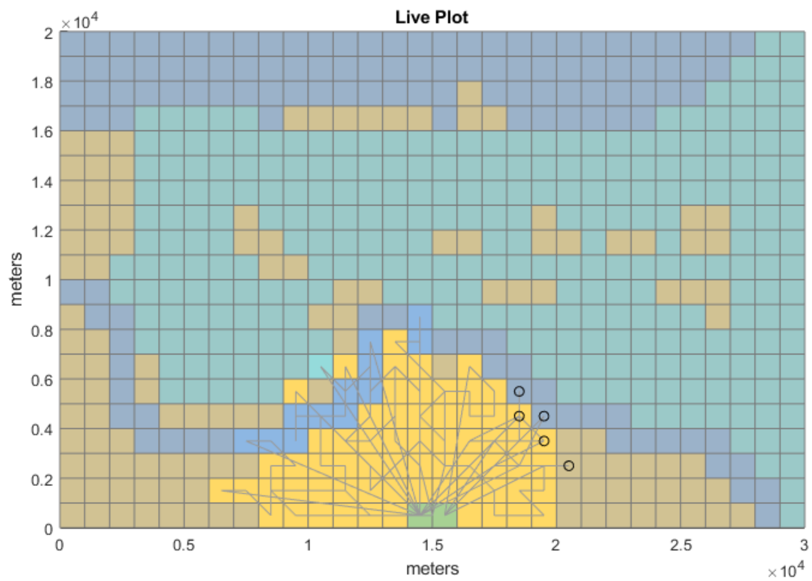

Figure 5.5.. The beginning of a 10 agent simulation with no coordination. Some agents are at the same location due to lack of coordination.

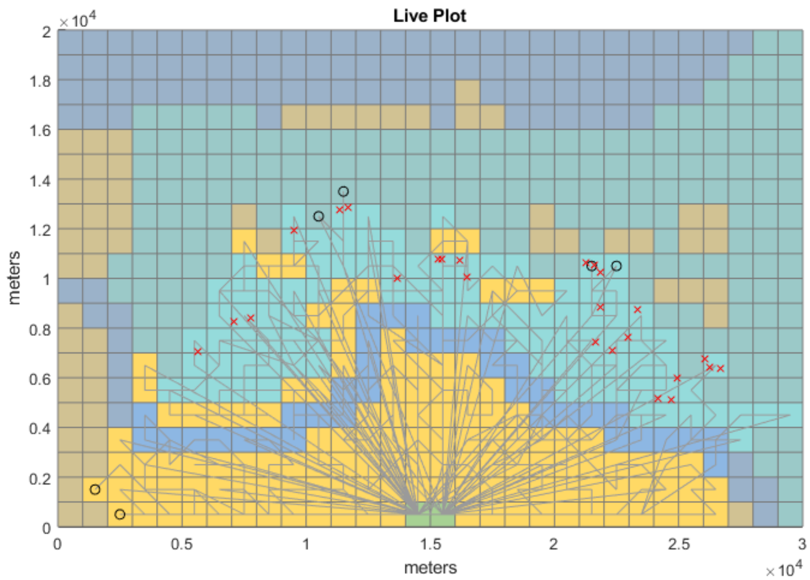

Figure 5.6.. Partway through a 10 agent simulation with no coordination. Some agents are at the same location due to lack of coordination. Note the red x's for located refugees.

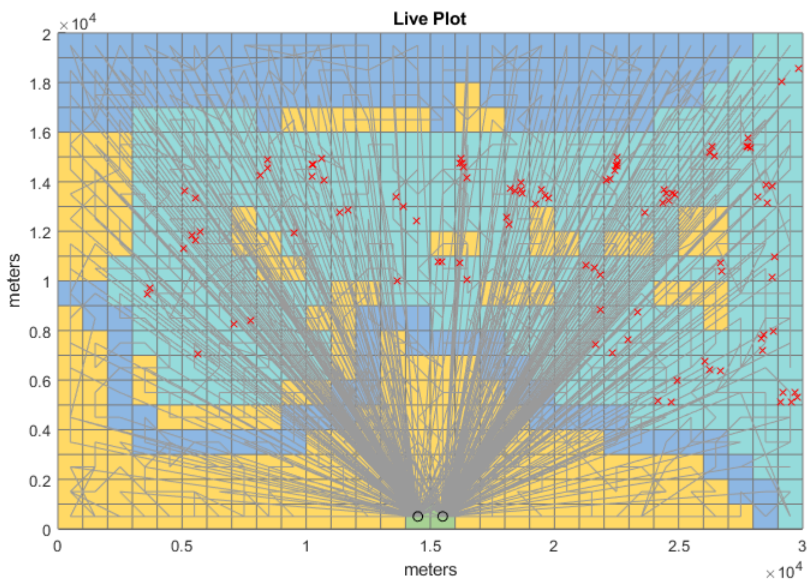

Figure 5.7.. Fully explored map at the end of a 10 agent simulation with no coordination.

5.3.1 Trend Analysis

Using this simulation, a parameter sweep can be performed to observe trends across these parameters. For this case study, a parameter sweep is done for all permutations of the parameters shown in table 5.11. Each permutation is run ten times due to the probabilistic nature of the problem. This results in a total of 240 runs. Results of these runs can be seen in figure 5.8. The raw data generated from these runs can be found in Appendix D.

Table 5.11.. Parameters to be tested in this Case Study.

Parameter	**Values**
Number of Agents	2, 5, 8, 10, 12, 15, 20
Maximum Retaskings	0, 1, Infinite

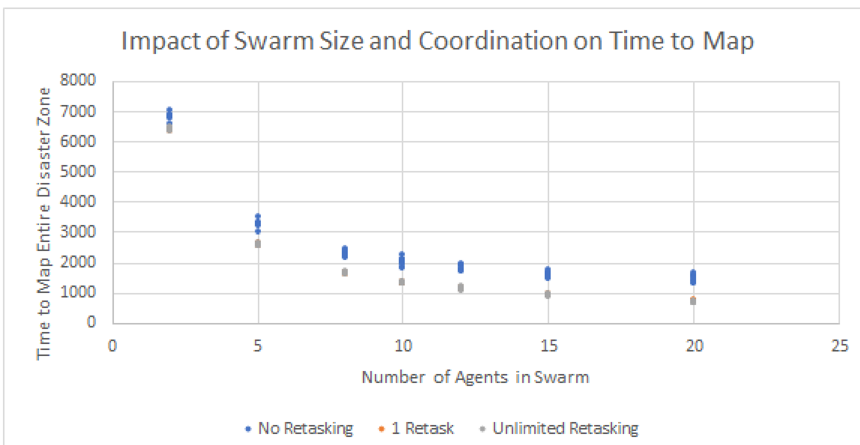

Figure 5.8.. Plot of mapping times for each parameter permutation.

Based on these results, some behavior trends can be observed. As the number of agents increases, the less time it takes to fully explore a set area, following a rational trend in all three coordination cases. This trend is not usual as the number of tiles is fixed, meaning that these tiles are going to be divided amongst the agents of the swarm. In addition, just adding one retasking cycle can reduce the time needed to explore a set area with the time saved increasing as the number of agents grow, but any more retasking cycles did not make much of a difference. The growing different between no coordination and some coordination can be attributed to there being less of a chance for agents to choose the same destination with smaller swarms. The runs without retasking also have a much larger spread in mapping times. This is likely due to there being more agents seeking out the same tiles and wasting time in the process at different points and frequencies in each run.

To understand why there is little difference between one retask compared to unlimited retasks, the actual runs need to be observed. Watching an animation of runs with no retasking, agents tended to group up and move along similar or identical paths, which results in similar timing between agents as they moved from tile to tile. Runs with a retasking option however did not have this, and as a result, agents moved different distances from each other, which effectively "desynced" these agents from each other, which prevented agents from choosing the same locations as each other. As a result, it seemed one retasking is the maximum needed in almost all cases.

Observing the animations also seemed to show some behavioral trends as well. Early in the simulation when unexplored tiles are closer to recharging stations, agents can explore several tiles before needing to recharge, creating the square paths seen in figure 5.5. Over time however, as agents need to move farther out, lines radiating from the recharging stations began to emerge as agents began to travel farther to find unexplored locations. At this point, agents are still able to explore multiple tiles before needing to return, but not as many as before. Towards the end, agents are only able to explore one or two tiles before needing to recharge as seen in figure 5.7. In addition, the radial patterns become more prominent, with noticeable breaks along the axis between the recharge stations. These initial observations show signs of *Scaling* in both time and space. As time went on, the paths began to change from mostly square paths between adjacent tiles to radial paths to move farther out. These corresponded to the radius of coverage.

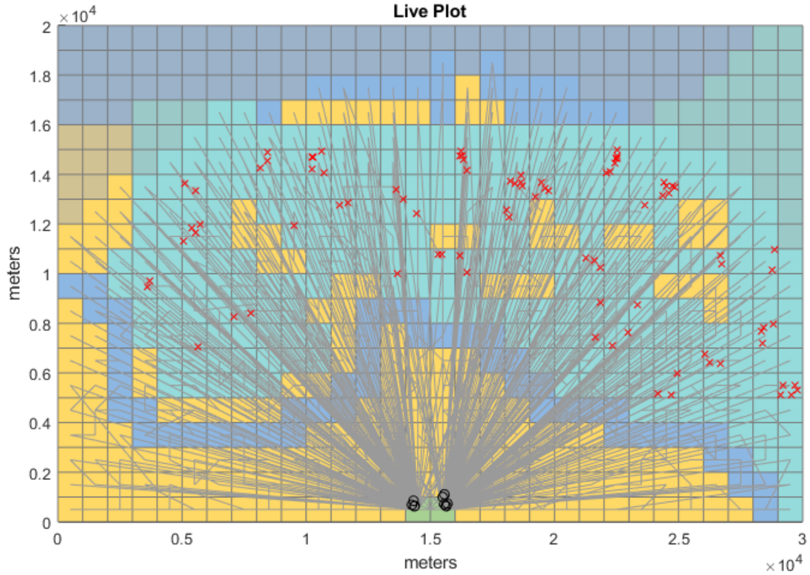

Figure 5.9.. Example run with agent battery life reduced to 40 minutes.

5.3.2 Qualitative Behavioral Study

Useful behavioral trends demand multiple runs, to give values that can be plotted and studied. This is not always necessary when trying to observe swarm behavior from a qualitative standpoint however. Basic behavior can be obtained just from modifying other agent or swarm parameters within a run. For instance, agents are currently assumed to have a maximum battery life of one hour. Changing this to forty minutes can be useful to judge how much battery life can be reduced to save weight ($Power$). Performing a run with this change led to the run seen figure 5.9. Immediately, it can be seen that there are many more radial lines compared to the square paths seen in figure 5.7. Not only that, agents are unable to explore the map as they did not have enough power to move to a destination, explore it, and return.

Meanwhile, extending the amount of space an agent explores within a time step from one tile to include the four adjacent tiles (five total tiles) can drastically reduce the time spent mapping (*Sensing*). Five minutes spent scanning a square kilometer can potentially be too much time. Changing this results in the scenario seen in figure 5.10, where the entire map is explored in 740 minutes by ten agents. Since each agent covers more ground, the paths are much more separated. Notably, this effect would likely be achieved through better sensors on the agents as well.

Changing recharge sites also have a pronounced impact on swarm behavior (external parameters). Shifting the sites north by a few kilometers and adding two more to create a square as shown in figure 5.11 ends up speeding up the mapping, reducing mapping time to just over 900 minutes. With a more central location, agents do not have to travel as fair to reach the edge of the map. In addition, more breaks begin to emerge along the axes between sites as well. A similar behavior can be seen when the sites are scattered like in figure 5.12. Radial lines are defined by the position of recharging stations, and spaces between them are predominantly made up of square paths.

—

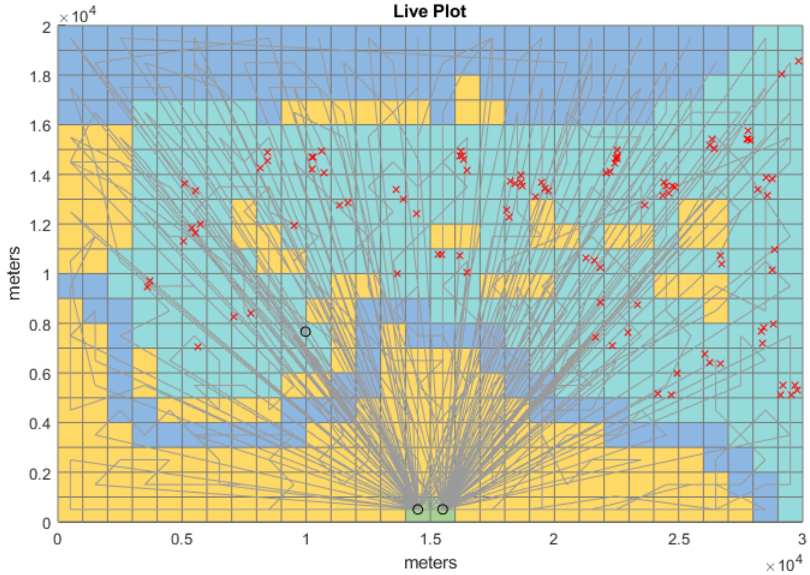

Figure 5.10.. Example run where agents can explore more space in a time step.

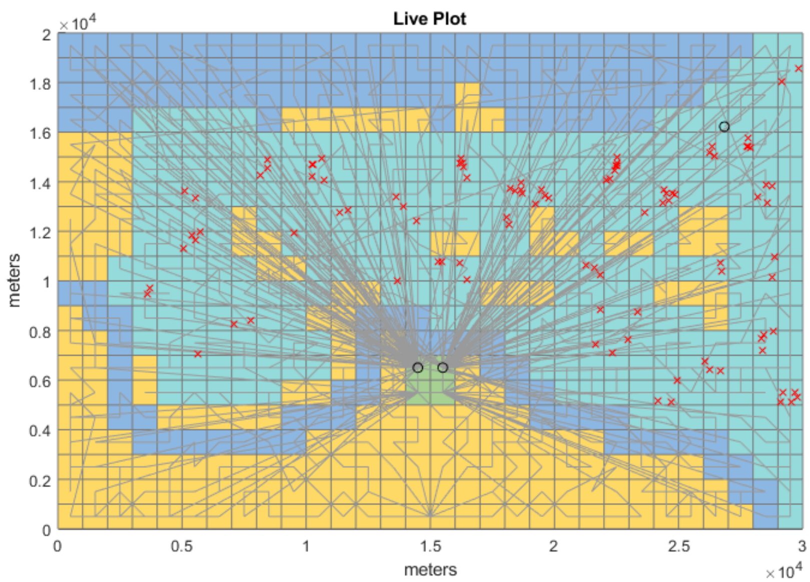

Figure 5.11.. Example run where recharge locations are modified.

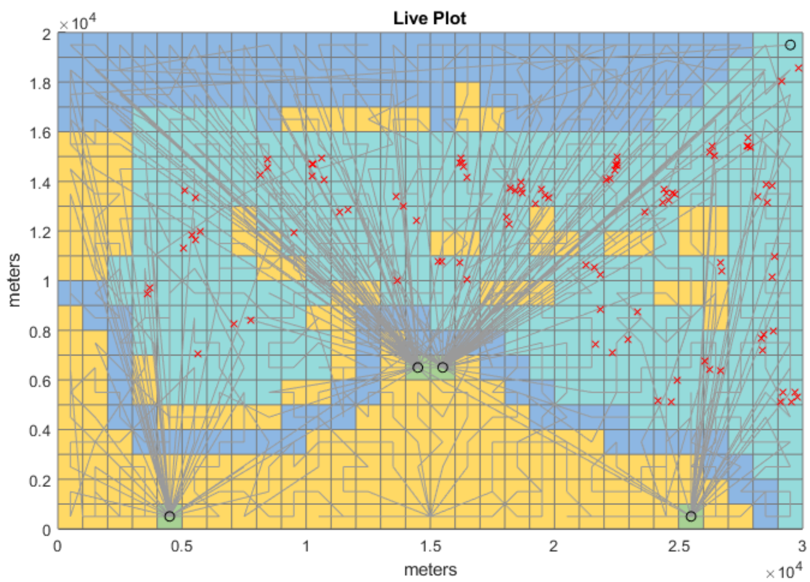

Figure 5.12.. Another example run where recharge locations are modified.

Overall, this case study has demonstrated the use of the Swarm Analysis Framework. A complex problem is analyzed and broken down into its components, allowing simpler subproblems to be studied. Meanwhile, relevant swarm characteristics are systematically identified, which are later used to develop a conceptual swarm design that can be tested to solve one of the subproblems (*Blind Search*). A deeper understanding provided by the Taxonomy of Swarm Characteristics allowed for simplifications to be made without removing key elements of the swarm, which reduced the complexity of the simulation. In addition to setting up parameter sweeps to show trends in swarm behavior, these characteristics also showed potential parameters that can be adjusted to create new behaviors.

From this point, the case study can be continued in several ways. Additional parameter sweeps can be performed with battery capacity or sensor range to explore trends with these parameters. Characteristics that are ignored can be factored back in to observe how swarm behavior changes as a result. An entirely different swarm architecture with hierarchies and heterogeneous agents can be developed and tested. The resolution of the map and timestep can also be reduced to better model short timescale behaviors. Different agent algorithms can be developed and tested. Study simplifications like the nature of refugees can be lifted, allowing for more complex tasks to be studied like *Patrolling* or *Source Search*. A physical realization of the swarm can even be developed at this point (though this would be a poor idea due to the limited knowledge gained at this point). The potential work from this point is endless, like how the Swarm Analysis Framework guided the specific study seen in this case study, it can also be used to guide any one of these future paths.

6. SUMMARY

The primary goal of this work is to establish a framework for defining a swarm and its accompanying characteristics and applications. With growing interest in swarm applications across multiple fields, it is more important than ever to develop a systematic tool for guiding design and analysis of such a complex topic. This is achieved through the Swarm Analysis Framework, which is described extensively in Chapters 3 and 4. In summary, the Swarm Analysis Framework is an extensive organizational approach that systematically analyzes different aspects of a swarm and its applications. The Taxonomy of Swarm Characteristics simultaneously captures the many parameters that define a swarm while acknowledging the many ways these interact and influence each other. The idea of organizing by scope also helps separate these parameters into clearly defined groups that directly pertain to a certain scope and aspect, which can be explore in greater detail as needed. Meanwhile, the Taxonomy of Swarm Applications can be used to decompose all manner of complex applications and goals into simple tasks and objectives that can be studied individually or collectively as needed. Such a breakdown is key for systematically understanding the causes of different swarm behaviors, an impossible task when looking at the combined effects of countless interactions.

A secondary goal of this work is to show how such a approach can be used to guide design and development of swarms. This is made possible by the generalized nature of the framework. As seen in the various examples in Chapter 3, this framework can be used to analyze swarms from multiple perspectives. From an agent-centric standpoint, it can be used to help design agents specifically for an architecture or evaluate if existing agents would suffice. It can also be used to identify possible swarm architectures and applications that can be achieved by an existing group of agents. Studying interactions between agents can help tune agent relationships to create more optimal behaviors or understand the mechanisms that drive existing swarms of living creatures, which can later be mimicked by artificial swarms. From a swarm level overview, agent requirements can be identified based on demands of *Group* level choices. With so many parameters, artificial intelligence and machine learning methods can also be used to develop better swarms by way of evolution. With such a variety of perspectives, all manner of design methodologies can benefit from the Swarm Analysis Framework.

As shown by the case study in Chapter 5, the framework can also guide design and testing. After decomposing applications and identifying key characteristics, informed simplifications can be made to narrow problem down to a manageable starting point that can be modeled and studied to gain initial insights in behavior, before gradually removing these simplifications to develop realistic solutions that can be built and used to solve real life problems.

The last goal of this work is to show how such a approach can be used within aerospace. This is shown in Chapters 4 and 5, where aerospace swarm applications analyzed, and a specific example is studied. Each aerospace application described can similarly be analyzed using the framework to understand the problem, identify requirements, develop swarm architectures, and guide analysis of swarm behavior. Despite the focus on aerospace however, the framework is just as capable of analysis swarm problems in robotics, biology, or even human interactions if the necessary characteristics are factored in.

Despite its benefits, it is also important to note that the Swarm Analysis Framework is intended to be an analysis approach that guides swarm understanding and development and does not outline or endorse a particular swarm design methodology. This framework also is not an optimization tool, though it can certainly be used to guide optimization attempts. Optimization is heavily dependent on application, so focusing on it would run counter to the goal of developing a framework for swarms and other multi-agent systems. Due to the general nature of the framework, it is also not comprehensive. It certainly strives to cover as much relevant information as possible but the design space for swarms is simply too large to capture every detail. By limiting its scope to single swarms, there is a vast number of situations that the framework is unable to presently cover. Despite this, it is the first and most extensive exploration of swarm characteristics and applications to date and is a critical starting point from which more complexity can be added. Should anything more specific be desired, the scope can be extended using the same principles that established the levels in the Taxonomy of Characteristics. Similarly, domain specific characteristics, objectives, or methods can be included as needed. In the end, the Swarm Analysis Framework is a well-defined starting point for which much more can be developed.

7. FUTURE WORK AND SUMMARY OF CONTRIBUTIONS

7.1 Next Steps

Despite the extensiveness of the Swarm Analysis Framework, there are still many areas it can be improved in order to better answer the question, "What is a swarm?" Many areas were not studied in order to maintain scope. Any one of the following areas can be expanded to improve the framework, which is by no means complete.

7.1.1 Deeper Discussion of Different Disciplines

As it stands, there is still missing context at the *Group* level of characteristics. Swarm intelligence, network theory, and multi-agent control are all rich disciplines that are only briefly discussed in Chapter 3. There are a great many principles within these domains that can be included to further show how different characteristics come together to create emergent, swarm defining behavior. In addition to identifying critical junctions in the swarm network, ideas within network theory can be used to identify inherent behaviors that result from a particular class type of network structure. Deeper understanding of swarm intelligence and multi-agent control can provide ways to better craft algorithms and swarm operations and evaluate their principle behaviors quickly. Including principles from fields like political science, economics, sociology, and psychology can also be used to address problems involving agents with more complex *BDI* or swarms that function off internal competition.

In addition, further exploration of system-of-systems methodologies and complexity theory can be used to refine the organizational structure of the framework to allow more seamless integration with existing tools. Complexity theory can be used to show more subtle links between characteristics and applications, while also help manage the numerous interactions and dependencies within a swarm. Principles of SoS can be used to show where changes are best made to gain the most benefit, while keeping the swarm operational against different types of disruptions. SoS Analysis tools can also be used to evaluate dependencies using the System Operational Dependency Analysis (SODA) method [91] or cost using Robust Portfolio Optimization (RPO) [92].

7.1.2 Extending Scope

In the future, the scope of the Swarm Analysis Framework can be expanded as well. As it stands, extending the scope to areas below the *Individual* level would be wholly unnecessary given the extensive body of knowledge in existence for all many of agent subsystems and components. Extending the the other direction holds promise. Beyond the *Group* level, impacts from external entities can start being addressed. The level immediately above *Group* would start addressing how a single swarm would interact with agents of another deliberative entity with different goals like an independent agent or an entirely separate swarm. Disciplines that involve cooperation and competition like game theory and economics can be readily used to explore relationships between distinct entities. Ideas from a military and defense standpoint can be explored in much better detail as these are predicated on the existence of another deliberative entity. Moving up further will bring the level that addresses simultaneously interacting swarms, where the interactions of multiple independent entities make SoS methodologies more relevant than ever. It is at this point that the highly relevant swarm problem of traffic management becomes possible, so extending the scope of the framework at least to this level is a highly desirable goal.

Alternatively, the scope can be expanded to better address the environment in which a swarm functions. The framework currently does not address the environment in any significant detail aside from acknowledging the basic limits it poses on a swarm. An improved understanding of the environment can provide better context for swarm operation. Better predictive models would help make a swarm more resilient to disruptions, and familiarity with a type of environment can be leveraged to reduce the workload of an agent or the entire swarm.

7.1.3 Exploring Swarm Design Methodologies

From a design standpoint, the framework lacks discussion on the many different ways swarms can be designed and the methods used to accomplish tasks. Currently the framework is able to provide a starting and end point for swarm design but possesses little on how to connect these two. Inclusion of swarm design would connect these two concepts, allowing the framework to become a full design suite. Knowledge of general design principles and swarm problem solving methods would also provide context on how to best connect a characteristic to an application, which can be used to develop different swarm archetypes and identify their best use cases.

7.1.4 Case Study

On a much smaller scale, the work from the case study in Chapter 5 can also be continued along any of the paths described at the end of the chapter. This does not have much bearing on the potential directions for the overall framework, but the disaster response problem is still one being actively studied, and future insights on that specific application can be gained if the case study is explored in greater detail.

7.2 Contributions to Swarm Research

Even with its limitations, this study and the Swarm Analysis Framework provides advancement in several areas of swarm research. The framework is the first of its kind, combining the multiple organizational tools proposed over the years into a single extensive and unified approach that address two of the least defined aspects of swarm design. It builds upon taxonomies developed to organize swarms within a single area of research and expands it to include factors across multiple disciplines. The ideas of organizing characteristics by scope and decomposing applications into components are both novel methods that bring well defined structures to highly complex aspects of swarm design.

Perhaps one of the more significant contributions is how the framework acts as a useful body of knowledge that brings together a staggering number of concepts from multiple disciplines and shows how each contribute to a swarm. Swarm intelligence, multi-agent control, network theory, and machine learning are all brought together alongside system-of-systems principles, modeling and simulation, biology, and system dynamics. Despite how many different ideas are already included, the framework is flexible enough to include even more specific areas as needed like aerodynamics, sensor design, or even theories of intelligence. No single concept is presented as the only way to approach a problem, and as a result, is a remarkably good starting point to begin exploring regardless of how much requisite knowledge a designer has to begin with.

This is potentially one of the earliest works compiling applications in aerospace that are not restricted to military applications for both aircraft and spacecraft. By no means is the one provided exhaustive, but there is little work available that contains all of the ones described in the latter section of Chapter 4. Most existing work is focused on a particular use case for aircraft or spacecraft or are focused on the design considerations of a particular problem. Few explore the potential applications across different environments, and even fewer compare and contrast these with each other.

With such a growing interest in utilizing swarms to solve complex problems, it is more important than ever to develop a unified approach to swarm design such that methods from one domain can easily be used to solve a similar problem in another. There is, of course, the perennial problem of creating a new standard only to contribute to an increasing number of conflicting standards [93], but at the very least, the hope is that the work presented in this study is a useful step in understanding how the humble swarm can be used to address at least one of the many problems that plague this world.

REFERENCES

REFERENCES

[1] Definition of Swarm by Oxford Dictionary. https://www.lexico.com/en/definition/swarm.

[2] Definition of Swarm by Merriam-Webster. https://www.merriam-webster.com/dictionary/swarm.

[3] Luca Iocchi, Daniele Nardi, and Massimiliano Salerno. Reactivity and Deliberation: A Survey on Multi-Robot Systems. In Markus Hannebauer, Jan Wendler, and Enrico Pagello, editors, *Balancing Reactivity and Social Deliberation in Multi-Agent Systems*, Lecture Notes in Computer Science, pages 9–32, Berlin, Heidelberg, 2001. Springer.

[4] Bruce T. Clough. UAV Swarming? So What are Those Swarms, What are the Implications, and How Do We Handle Them? Technical Report AFRL-VA-WP-TP-2002-308, AIR FORCE RESEARCH LAB WRIGHT-PATTERSON AFB OH AIR VEHICLES DIRECTORATE, April 2002.

[5] Levent Bayindir and Erol Sahin. A Review of Studies in Swarm Robotics. *Turkish Journal of Electrical Engineering*, 15(2), 2007.

[6] Gerardo Beni. Swarm Intelligence. In Robert A. Meyers, editor, *Encyclopedia of Complexity and Systems Science*, pages 1–28. Springer, Berlin, Heidelberg, 2019.

[7] Roberto A. Flores-Mendez. Towards the Standardization of Multi-Agent Systems Architectures: An Overview. *Acm Crossroads Student Magazine*, 5:18–24, 1999.

[8] John Arquilla and David Ronfeldt. *Swarming and the Future of Conflict:*. RAND Corporation, 2000.

[9] G. Dudek, M. Jenkin, E. Milios, and D. Wilkes. A taxonomy for swarm robots. In *Proceedings of 1993 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '93)*, volume 1, pages 441–447 vol.1, July 1993.

[10] Gregory Dudek, Michael R. M. Jenkin, Evangelos Milios, and David Wilkes. A taxonomy for multi-agent robotics. *Autonomous Robots*, 3(4):375–397, December 1996.

[11] Y. Uny Cao, Alex S. Fukunaga, and Andrew Kahng. Cooperative Mobile Robotics: Antecedents and Directions. *Autonomous Robots*, 4(1):7–27, March 1997.

[12] M. Lindauer. Communication in Swarm-Bees Searching for a New Home. *Nature*, 179(4550):63–66, January 1957.

[13] S. Goss, S. Aron, J. L. Deneubourg, and J. M. Pasteels. Self-organized shortcuts in the Argentine ant. *Naturwissenschaften*, 76(12):579–581, December 1989.

[14] Alberto Colorni, Marco Dorigo, and Vittorio Maniezzo. Distributed Optimization by Ant Colonies. In *Proceedings of the First European Conference on Artificial Life*, January 1991.

[15] Deborah Gordon. The organization of work in social insect colonies. *Nature*, 380:121–124, March 1996.

[16] Craig W. Reynolds. Flocks, herds and schools: A distributed behavioral model. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '87, pages 25–34, New York, NY, USA, August 1987. Association for Computing Machinery.

[17] Levent Bayindir. A review of swarm robotics tasks. *Neurocomputing*, 172:292–321, January 2016.

[18] Yogeswaran Mohan and S. G. Ponnambalam. An extensive review of research in swarm robotics. In *2009 World Congress on Nature Biologically Inspired Computing (NaBIC)*, pages 140–145, December 2009.

[19] Iñaki Navarro and Fernando Matía. An Introduction to Swarm Robotics. https://www.hindawi.com/journals/isrn/2013/608164/, 2013.

[20] Tucker Balch. Taxonomy of Multirobot Task and Reward. *Robot teams: From diversity to polymorphism*, March 1999.

[21] Paul Scharre. Unleash the Swarm: The Future of Warfare. https://warontherocks.com/2015/03/unleash-the-swarm-the-future-of-warfare/, March 2015.

[22] Mark W. Maier. Architecting principles for systems-of-systems. *Systems Engineering*, 1(4):267–284, 1998.

[23] Daniel DeLaurentis. Understanding Transportation as a System-of-Systems Design Problem. In *43rd AIAA Aerospace Sciences Meeting and Exhibit*. American Institute of Aeronautics and Astronautics, 2005.

[24] Sarah Sheard and Dr Mostashari. A Complexity Typology for Systems Engineering. *INCOSE International Symposium*, 20, July 2010.

[25] Mark Newman. *Networks*. Oxford university press, 2018.

[26] M. E. J. Newman. The Structure and Function of Complex Networks. *SIAM Review*, 45(2):167–256, January 2003.

[27] United States. Joint Doctrine Capstone Publications. Technical report, U.S. Joint Chiefs of Staff, March 2013.

[28] Daniel Sukman. The Institutional Level of War. https://thestrategybridge.org/the-bridge/2016/5/5/the-institutional-level-of-war.

[29] Martin Dunn. Levels of War: Just a Set of Labels? Newsletter, Directorate of Army Research and Analysis, Australia, October 1996.

[30] Michael R. Matheny. The Fourth Level of War. http://ndupress.ndu.edu/Media/News/News-Article-View/Article/643103/the-fourth-level-of-war/, January 2016.

[31] Sarah A. Sheard and Ali Mostashari. Principles of complex systems for systems engineering. *Systems Engineering*, 12(4):295–311, 2009.

[32] Russell Lincoln Ackoff. *Ackoff's Best: His Classic Writings on Management*. Wiley, New York, 1999.

[33] David D. Walden, Garry J. Roedler, Kevin Forsberg, R. Douglas Hamelin, and Thomas M. Shortell. *Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities*. INCOSE, fourth edition, 2015.

[34] Alan Faisandier and Garry Roedler. System Architecture - SEBoK. *The Guide to the Systems Engineering Body of Knowledge (SEBoK)*, May 2020.

[35] Alan Faisandier and Rick Adcock. Physical Architecture Model Development - SEBok. *The Guide to the Systems Engineering Body of Knowledge (SEBoK)*, May 2020.

[36] Alan Faisandier and Garry Roedler. Logical Architecture Model Development - SEBoK. *The Guide to the Systems Engineering Body of Knowledge (SEBoK)*, May 2020.

[37] Benjamin S. Blanchard and Wolter J. Fabrycky. *Systems Engineering and Analysis*. Pearson, fourth edition, 2006.

[38] Nigel Cross. *Engineering Design Methods: Strategies for Product Design*. John Wiley & Sons, Chichester, England, 3rd edition, 2000.

[39] Ulrich Nehmzow. *Mobile Robotics: A Practical Introduction*. Springer-Verlag, London, second edition, 2003.

[40] Jeffrey M. Bradshaw. *An Introduction to Software Agents*. AAAI Press, 1997.

[41] Michael Bratman et al. *Intention, Plans, and Practical Reason*, volume 10. Harvard University Press Cambridge, MA, 1987.

[42] Michael Georgeff, Barney Pell, Martha Pollack, Milind Tambe, and Michael Wooldridge. The Belief-Desire-Intention Model of Agency. In Jörg P. Müller, Anand S. Rao, and Munindar P. Singh, editors, *Intelligent Agents V: Agents Theories, Architectures, and Languages*, Lecture Notes in Computer Science, pages 1–10, Berlin, Heidelberg, 1999. Springer.

[43] DJI. Mavic Air 2- Specifications - DJI. https://www.dji.com/ca/mavic-air-2/specs.

[44] Wikipedia Contributors. Starlink. *Wikipedia*, June 2020.

[45] SpaceX. Starlink. http://www.starlink.com.

[46] Stephen Clark. SpaceX releases new details on Starlink satellite design – Spaceflight Now.

[47] Bernat Corominas-Murtra, Joaquín Goñi, Ricard V. Solé, and Carlos Rodríguez-Caso. On the origins of hierarchy in complex networks. *Proceedings of the National Academy of Sciences*, 110(33):13316–13321, August 2013.

[48] Zhong Thai, Prajwal Balasubramani, Chris Brand, Andrew Haines, and Daniel DeLaurentis. PLEXNet – A Distributed, Variable Autonomy Architecture for Exploration of Planetary Bodies. In *IAF Space Exploration Symposium*, Washington D.C., 2019.

[49] Zhong W. Thai, Prajwal Balasubramani, Chris Brand, Andrew Haines, and Daniel A. DeLaurentis. Study of Swarm-based Planetary Exploration Architectures Using Agent-Based Modeling. In *AIAA Scitech 2020 Forum*. American Institute of Aeronautics and Astronautics, January 2020.

[50] Wikipedia Contributors. Ant. *Wikipedia*, July 2020.

[51] Paul Baran. On Distributed Communications. Memorandum RM-3420-PR, The Rand Corportaion, 1964.

[52] MaidSafe. Evolving Terminology with Evolved Technology: Decentralized versus Distributed. https://medium.com/safenetwork/evolving-terminology-with-evolved-technology-decentralized-versus-distributed-7f8b4c9eacb, July 2018.

[53] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1):269–271, December 1959.

[54] Leonid E. Zhukov. Centrality Measures, February 2016.

[55] Paolo Boldi and Sebastiano Vigna. Axioms for Centrality. *arXiv:1308.2140 [physics]*, November 2013.

[56] Wikipedia Contributors. Centrality. *Wikipedia*, December 2019.

[57] Albert-Laszlo Barabasi and Eric Bonabeau. Scale-Free Networks. *Scientific American*, 288(5):60–69, 2003.

[58] Stafford Beer. Diagnosing the system for organizations. 1985. *Great Britain: John Wiley and Sons Ltd*, 1995.

[59] Tim Hartnett. *Consensus-Oriented Decision-Making: The CODM Model for Facilitating Groups to Widespread Agreement*. new society publishers, 2011.

[60] Wikipedia Contributors. Artificial neural network. *Wikipedia*, June 2020.

[61] Jay W. Forrester. Industrial Dynamics. https://www.questia.com/library/408987/industrial-dynamics, 1961.

[62] John Boyd. *Destruction and Creation*. US Army Comand and General Staff College, 1987.

[63] Frédéric Delsuc. Army Ants Trapped by Their Evolutionary History. *PLoS Biology*, 1(2), November 2003.

[64] M. B. Quadrelli, J. Chang, E. Mettler, W. Zimmermann, S. Chau, and A. Sengupta. System architecture for guided herd of robots exploring Titan. In *2004 IEEE Aerospace Conference Proceedings (IEEE Cat. No.04TH8720)*, volume 1, pages 379–387 Vol.1, March 2004.

[65] Michio Sugahara and Fumio Sakamoto. Heat and carbon dioxide generated by honeybees jointly act to kill hornets. *Naturwissenschaften*, 96(9):1133–1136, September 2009.

[66] Krishnendu Mukopadhyaya. Distributed Algorithms for Swarm Robots, February 2015.

[67] Evan Ackerman. Swarm of Robots Forms Complex Shapes Without Centralized Control - IEEE Spectrum. https://spectrum.ieee.org/automaton/robotics/robotics-hardware/swarm-of-robots-forms-complex-shapes-without-centralized-control, March 2020.

[68] Reka Albert and Albert-Laszlo Barabasi. Statistical mechanics of complex networks. *Reviews of Modern Physics*, 74(1):47–97, January 2002.

[69] Wayne Bailey, Cormac Cannon, and Brandt Payne. *The Complete Marching Band Resource Manual: Techniques and Materials for Teaching, Drill Design, and Music Arranging*. University of Pennsylvania Press, March 2015.

[70] Eric Bonabeau. Swarm Intelligence, April 2003.

[71] J. Kennedy and R. Eberhart. Particle swarm optimization. In *Proceedings of ICNN'95 - International Conference on Neural Networks*, volume 4, pages 1942–1948 vol.4, November 1995.

[72] Olivier L. De Weck, Daniel Roos, and Christopher L. Magee. *Engineering Systems: Meeting Human Needs in a Complex Technological World*. MIT Press, 2011.

[73] James D. Willis and Steven Dam. The Forgotten "-ilities", 2011.

[74] Garret N. Vanderplaats. *Multidiscipline Design Optimization*. VR&D, January 2007.

[75] Jamey D. Jacob, Phillip B. Chilson, Adam L. Houston, and Suzanne Weaver Smith. Considerations for Atmospheric Measurements with Small Unmanned Aircraft Systems. *Atmosphere*, 9(7):252, July 2018.

[76] Leoncio Briones, Paul Bustamante, and Miguel A. Serna. Wall-climbing robot for inspection in nuclear power plants. In *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, pages 1409–1414 vol.2, May 1994.

[77] Praveen Sekhar and Reshmi Bhooshan. Duct fan based wall climbing robot for concrete surface crack inspection. In *2014 Annual IEEE India Conference (INDICON)*, pages 1–6, 11.

[78] mndotresearch. Using drones to inspect bridges, September 2015.

[79] Paul Beardsley. VertiGo - a Wall-Climbing Robot including Ground-Wall Transition, 2015.

[80] Jae-Uk Shin, Donghoon Kim, Jong-Heon Kim, and Myung Hyun. Micro aerial vehicle type wall-climbing robot mechanism. In *2013 IEEE RO-MAN*, pages 722–725, 26.

[81] Paolo Tripicchio, Massimo Satler, Giacomo Dabisias, Emanuele Ruffaldi, and Carlo Alberto Avizzano. Towards Smart Farming and Sustainable Agriculture with Drones. In *2015 International Conference on Intelligent Environments*, pages 140–143, July 2015.

[82] Brian Barrett. Inside the Olympics Opening Ceremony World-Record Drone Show. *Wired*, February 2018.

[83] Daniel Wassmuth and Dave Blair. Loyal Wingman, Flocking, and Swarming: New Models of Distributed Airpower. https://warontherocks.com/2018/02/loyal-wingman-flocking-swarming-new-models-distributed-airpower/, February 2018.

[84] A Poghosyan, I Lluch, H Matevosyan, A Lamb, C Moreno, C Taylor, A Golkar, Judith Cote, S Mathieu, Stephane Pierotti, J Grave, J Narkiewicz, S Topczewski, M Sochacki, Estefany Lancheros, Hyuk Park, Adriano Camps, and A Ru. Unified Classification for Distributed Satellite Systems. In *4th International Federated and Fractionated Satellite Systems Workshop*, Rome, Italy, October 2018.

[85] Adam Mann and 2019. What Is a Dyson Sphere? https://www.space.com/dyson-sphere.html, August 2019.

[86] Jekanthan Thangavelautham, Kenneth Law, Terence Fu, Nader Abu El Samid, Alexander D. S. Smith, and Gabriele M. T. D'Eleuterio. Autonomous multirobot excavation for lunar applications. *Robotica*, 35(12):2330–2362, December 2017.

[87] History.com Editors. Hurricane Katrina. https://www.history.com/topics/natural-disasters-and-environment/hurricane-katrina.

[88] LSU Katrina Survey Team. Post-Hurricane Katrina Research Maps. https://www.lsu.edu/faculty/fweil/KatrinaMaps/index.htm, 2008.

[89] Eric Bonabeau. Agent-based modeling: Methods and techniques for simulating human systems. *Proceedings of the National Academy of Sciences*, 99(suppl 3):7280–7287, May 2002.

[90] Robert Axtell. Why Agents? On the Varied Motivations for Agent Computing in the Social Sciences, November 2000.

[91] Cesare Guariniello and Daniel DeLaurentis. Dependency Analysis of System-of-Systems Operational and Development Networks. *Procedia Computer Science*, 16:265–274, January 2013.

[92] Navindran Davendralingam and Daniel A. DeLaurentis. A Robust Portfolio Optimization Approach to System of System Architectures. *Systems Engineering*, 18(3):269–283, 2015.

[93] Randall Munroe. Standards. https://xkcd.com/927/.

APPENDICES

A. SUMMARIZED TAXONOMY OF CHARACTERISTICS

This section summarizes general swarm characteristic categories and subcategories presented in the Taxonomy of Characteristics in Chapter 3 and provides examples for each of these. Note that the ones shown here are meant to show the types of characteristics that are captured by this taxonomy and is NOT exhaustive. Certain applications may require additional categories and subcategories as needed. These characteristics can also be used to help develop functions within a morphological chart.

Individual

Characteristics that apply to a single agent independent of others

Physical

Components and subsystems of an agent that describe the extent to which an agent can do something

- *Locomotion* – how agents move
  - terrain – aerial, aquatic, terrestrial, orbital, urban
  - mode of locomotion – wheels, legs, tracks, propellers
  - kinematics – linear and angular speeds
  - directional control – steering system, control surfaces, momentum wheels and gyroscopes
- *Manipulation* – how agents affect the world around them
  - target of manipulation – size, shape, materials
  - mode of manipulation – grabbers, pushers, suction

  - limitations – max/min manipulable values, temperatures

- *Sensing* – how agents acquire information
  - digital or analog
  - type of sensor – optical, pressure, temperature, electromagnetic
  - sensor ranges – sensor bands, saturation values
- *Transmission* – how agents transmit information
  - mode of transmission – broadcast, directed, wired
  - method of transmission – electromagnetic, audio, chemical, visual, motion, exchange of memory storage
- *Processing* – what agents do with information
  - software – operating systems, programs
  - specifications – processing speeds, number of cores, efficiency
- *Memory* – how agents store and recall information
  - mode of memory – paper printouts, electronic data storage, biological memory
  - amount of memory
  - data retrieval
  - data reliability
- *Power* – what enables an agent to continue functioning
  - method of storage – batteries, fuel cells, springs and flywheels, chemicals, starches
  - power generation – solar, petrochemical, kinetic, eating and digestion
  - efficiency, storage amount, power usage rates
- *Structure* – how components are arranged and with what materials
  - arrangement of components
  - materials used – plastic, metals, ceramics, composites, biological tissues

  - interference of components
  - agent modularity – compactable, fixed size, multiple configurations, interchangeable parts

Functional

Operations and algorithms that control an agent's components and subsystems

- *Motion* – how components create motion
  - path planning, when to use locomotion mechanisms
  - collision avoidance
- *Information Transfer* – how and when to communicate
  - receivable signals and information
  - transmittable signals and information
  - security – encryption
- *Control* – how an agent self regulates
  - dynamic response – stability, feedback loops
  - homeostasis
  - basic functions – when to use what components, how to use components
- *Disruption* – how agents react and adjust to changes
  - types of disruptions part failure, environmental change, collisions
  - reaction – change of goals, restoration of previous states, escape
  - backups – contingencies, backup components and subsystems
- *Tasking* – how goals are determined and prioritized
  - priority
  - order
- *Beliefs, Desires, Intentions* – why an agent makes a decision

  - beliefs – what the agent knows (knowledge)
  - desires – what the agent wants (goals)
  - intentions – how desires and goals are realized (actions and motivations)
- *Self Awareness* – is the agent aware of itself and its limitations
  - knowledge of own limitations
  - ability to recognize itself
  - self-preservation

Interaction

Characteristics that describe the relationship between any two agents

Awareness

If relationships can be made between two agents

- *Swarm Awareness* – if agents know they are part of a swarm and if agents can identify other agents in that swarm
  - knowledge that an agent is part of a swarm
  - knowledge of other agents in swarm through direct observation or second hand knowledge
  - ability to distinguish signal of peers from external signals
- *Environmental Awareness* – if agents are aware of the environment
  - ability to detect features of the environment
  - active use of this information to modify behavior
- *Prediction* – if agents can model and predict other agents or the environment
  - methods for prediction – finite state automata, neural network, response surface, classification trees, decision trees
  - reliability, robustness, and uncertainties

Communication

What characteristics define existing relationships

- *Transmission and Receiving* – how two agents are linked
  - receiving – type of signals used by each agent for receiving
  - transmission – type of signals used by each agent for transmission
  - type of signal – broadcast, directed, environmental (stigmergic)
- *Range, Bandwidth, Latency* – how much and how far can information be sent between two agents
  - range – maximum distance a line of communication can be maintained based on agents and environment
  - bandwidth – maximum transmittable information for a given agent and environment
  - latency – time required for transmission to reach target

Roles

How relationships are affected by differences between agents

- *Purpose* – what are an agent's goals and why does it exist
  - reason agent exists in a vacuum
  - reason agent exists within the swarm – sensing, relaying information, maintenance, processing, coordination
  - dependencies between agents
- *Rank* – whether an agent has the power to influence or control another
  - type of relationship – peer-to-peer, superior-subordinate, cyclical, conditional superior-subordinate
  - number of superiors and subordinates
  - strength of relationship

- *Cooperation* – how conflicts between two agents are resolved
  - based on relationship – defer to rank, expertise, specialization, purpose
  - voting, consult peers, random choice

Group

Characteristics that begin to exist with multiple interacting agents

Composition

Observable features of a swarm

- *Size* – how many agents are in a swarm or its subgroups
  - total agents in swarm
  - number of agents with set of characteristics
  - order of magnitude
- *Homogeneity* – how different are agents from each other
  - ratio of different types of agents in a swarm or subset of agents
  - ratio of agents with specific features in a swarm of subset of agents
- *Scaling* – can a swarm grow or shrink without significant changes to behavior
  - scaling breakpoints
  - types of changes - addition/removal of agents types, ratio changes or reversals, changes in the number of minority types

Capability

What a swarm is able to do

- *Access* – where a swarm's agents are able to go
  - terrain – aerial, aquatic, terrestrial, orbital, urban

  - access by size – small tunnels and holes, large cracks

- *Range* – how far can a swarm affect objects or the environment
- *Output* – what is the total output of a swarm
  - additive outputs – total force output, coverage
  - limited outputs – lifespan of swarm

Networks

The underlying structure created by relationships between agents at the interaction level

- *Connectivity* – how connected are agents to each other
- *Distribution* – where are connections between agents concentrated
- *Centrality* – what agents are critical to the swarm
  - betweenness centrality
  - closeness centrality
  - eigenvector centrality and degree centrality
  - page rank
  - hierarchical centrality – number of agents at highest rank
- *Structure* – what is the overall structure of the swarm network
  - network archetypes – scale free, small world, bipartite, strict hierarchy, hypergraph
- *Hierarchy* – what is the chain of command in the swarm
- *Physical Network* – how agents are physically arranged at a given moment
- *Organization* – how agents are grouped and how do these groups interact
  - role based – groups determined by roles and specialties
  - balanced – roles distributed among groups

  - regional – groups determined by proximity
  - rank based
  - dependency based

Intelligence

How a swarm processes information and makes decisions as a group

- *Consensus* – how do agents agree
  - defer to hierarchy
  - voting and other democratic methods
  - social methods
  - averaging or weighted ranking
- *Coordination* – how agents know what to do and when
  - purpose – tasking, path planning, process management
  - centralized methods – dependent upon dedicated controllers
  - distributed methods – local coordinators that work together or separately
  - indirect methods – control through interfaces and environment
  - development – logic and algorithms, developed through AL/ML
- *Group Computation* – are agents able to solve complex problems by combining their computation power
- *Learning* – are agents able to use past experiences to change behavior
  - methods for prediction – finite state automata, neural network, response surface, classification trees, decision trees
  - reliability, robustness, and uncertainties

Dynamics

How a swarm changes over time

- *Information Latency* – how do communication delays affect behavior
  - time to update all agents
  - time from between data acquisition and resulting decision (OODA loop)
- *Physical Latency* – how do delays impact physical changes and synchronization of the swarm
  - synchronization of agents
  - time to perform a coordinated and synchronized task
- *Stability* – what are the dynamic response characteristics and are there any stable or metastable states
  - stable and metastable behaviors
  - divergence
  - feedback loops – reinforcing, balancing, delay
- *Evolution* – how does a swarm's behavior change over time
  - learning based – behavior has been modified due to past experiences
  - environmental change based – area has been covered, environment changes significantly
  - task completion based – setup, standard operation, finishing

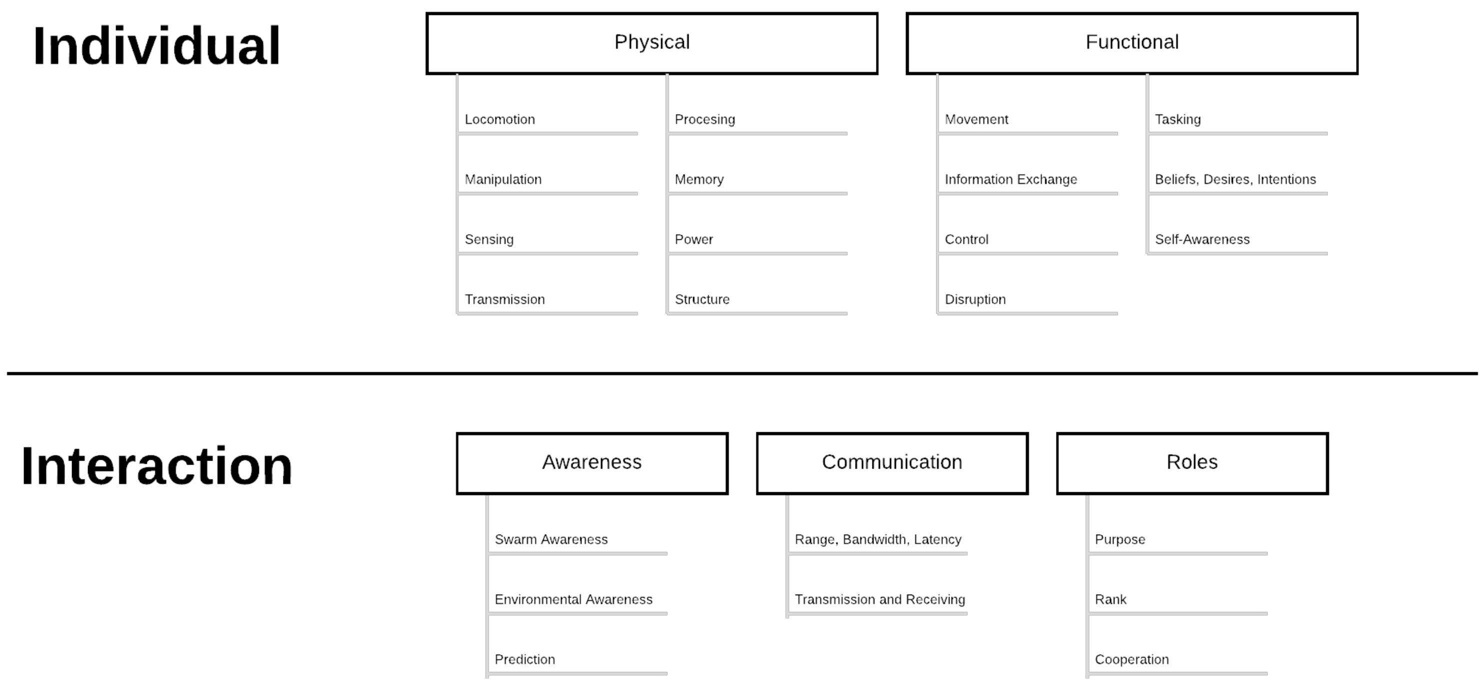

Figure A.1.. *Individual* and *Interaction* levels of the Taxonomy of Swarm Characteristics with Subgroups

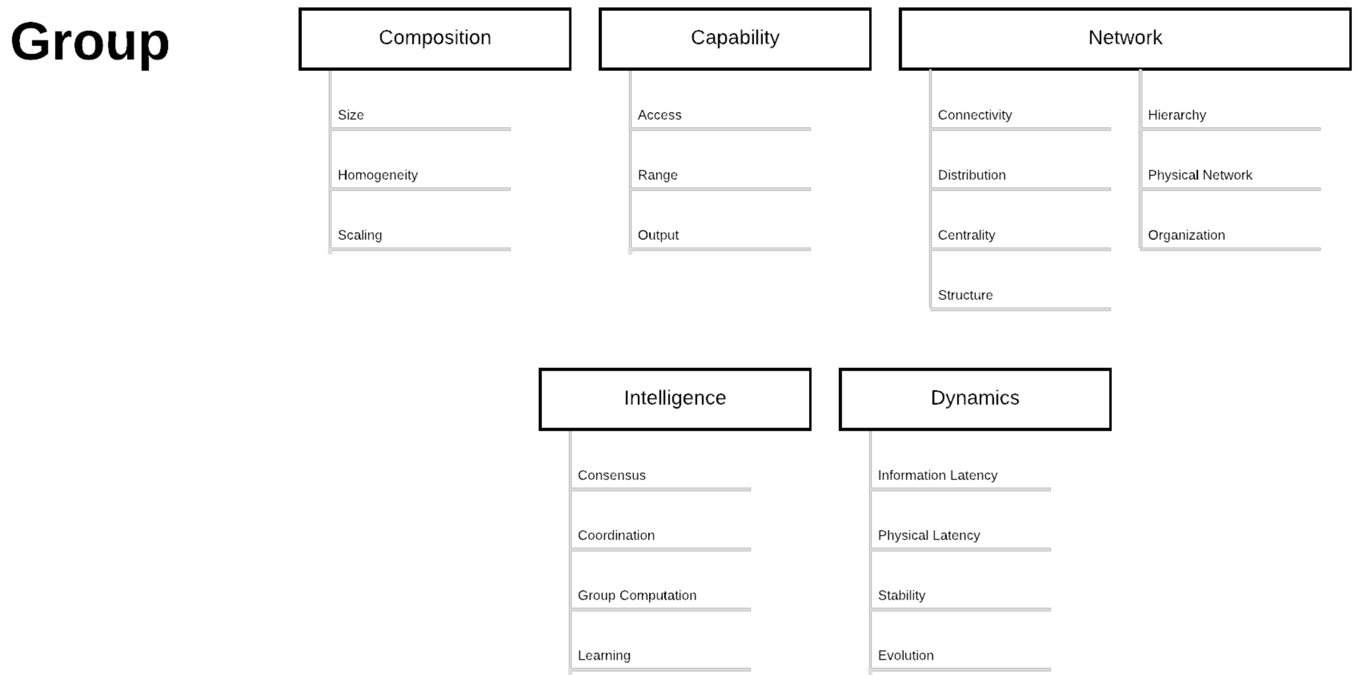

Figure A.2.. *Group* level of the Taxonomy of Swarm Characteristics with Subgroups

B. SUMMARIZED TAXONOMY OF APPLICATIONS

This section summarizes general swarm application components presented in the Taxonomy of Applications in Chapter 4. Note that the example Tasks and Objectives shown are NOT exhaustive and that more exist. The examples shown in the Tasks section may also feature elements of other tasks. These components may be combined in different ways to create highly complex applications.

Tasks

Aggregation

- Gathering – convergence of agents or objects towards one state or set of states
- Clustering – convergence of agents or objects towards multiple states based on varying criteria
- Flocking – dynamic convergence of agent or object states
- Collaborative Motion – gathering agents and coordinating a group to perform a monumental or complex task
- Overwhelm – gathering enough agents or objects to overwhelm a target

Dispersion

- Coverage – divergence of agent or object states to maximize coverage of one state or set of states
- Exploration – dynamic time based divergence where agents must avoid previously visited states as well

- Patrolling – dynamic time based divergence where the environment changes enough that agents need to revisit previous states
- Fractionated System – conceptual divergence where agent functions and states are as divergent as the swarm allows

Formation

- Relative Positioning – maintaining difference in states between agents based on a range
- Connectivity – maintaining connectivity requirements between agents or objects
- Shapes and Patterns – creating a highly defined shape or pattern with agents or objects
- Network Building – growing a formation based on a ruleset
- Coordinated Motion – moving a type of formation without distorting or disrupting the original formation
- Self-Assembly – type of formation where agents physically attach to each other based on a strict plan or ruleset
- Structure Building – type of formation where agents physically attach objects to each other based on a strict plan or ruleset

Search

- Blind Search – moving across a state or set of states to locate a target that meets a set of criteria
- Source Search – locating a target by tracking its emissions or signs
- Detection – stationary search where agents monitor an area until a target enters the swarm's range

- Emergent Search – emergent uncoordinated search developed using feedback loops in agent behavior

Objectives

- Efficiency (Cost) – optimizing the initial and operating costs of a swarm
- Efficiency (Resources) – optimizing the resource demands of a swarm
- Efficiency (Time) – optimizing the time required to perform a task or total lifecycle
- Adaptability – the swarm's ability to adjust to short-term and long-term changes
- Reliability – the swarm's ability to continue functioning as designed without failures
- Usability – the swarm's ease of use or integration into human controlled systems
- Flexibility – the swarm's ability to do multiple types of tasks
- Resilience – the swarm's resistance to damage or adverse conditions
- Scalability – the swarm's ability to grow without requiring significant changes in behavior
- Modularity – the ability to add or remove agents into a swarm

C. TASK AND OBJECTIVE BREAKDOWN OF AEROSAPCE APPLICATIONS

			Tasks				Objectives										Aggregation					Dispersion				Formation							Search			
			Aggregation	Dispersion	Formation	Search	Efficiency Cost	Efficiency Resources	Efficiency Time	Adaptability	Reliability	Usability	Flexibility	Resilence	Scalability	Modularity	Gathering	Clustering	Flocking	Collab. Manip.	Overwhelm	Coverage	Exploration	Patrolling	Fractionated System	Relative Positioning	Connectivity	Patterns and Shapes	Network Building	Coordinated Motion	Self-Assembly	Structure Building	Blind Search	Source Search	Detection	Path Formation
Air	Civilian	Communication Network		x	x		x				x											x		x		x	x		x							
		Aerial Mapping		x	x	x	x		x		x		x	x								x	x			x			x	x			x	x		
		Weather Monitoring		x	x	x				x	x	x	x	x	x	x						x		x		x		x	x					x	x	
		Disaster Search and Rescue		x	x	x			x	x	x	x	x	x	x	x						x	x	x			x		x			x	x	x	x	x
		Infrastructure Inspection				x	x				x	x		x	x	x																		x	x	x	
		Agricultural Applications		x	x	x	x				x	x	x	x	x	x						x		x			x			x			x	x	x	
		Displays and Shows			x		x	x			x	x		x												x		x	x	x						
	Military	Overwhelm	x		x		x	x									x	x	x		x					x		x								
		Distraction/Decoy	x		x			x		x	x					x	x	x	x		x							x		x						
		Surveillance		x	x	x	x				x			x	x							x		x			x		x					x	x	
		Strategic Formation		x	x				x	x	x			x								x				x		x								
		Wingman		x	x			x		x	x	x	x	x		x									x	x				x						

Figure C.1.. Extended mapping of aeronautical examples to the different types of *Tasks* and *Objectives* described in the Swarm Framework, including a mapping to specific application tasks.

			Tasks				Objectives									
			Aggregation	Dispersion	Formation	Search	Efficiency Cost	Efficiency Resources	Efficiency Time	Adaptability	Reliability	Usability	Flexibility	Reslience	Scalability	Modularity
Space	Constellations	Sensor Constellation		x	x	x	x	x			x			x	x	
		GPS Constellation		x	x		x	x			x	x		x	x	
		Communication Constellation		x	x		x	x			x			x	x	
		Power Transmission		x	x		x	x			x	x		x	x	
	Clusters	Fractionated Satellite		x	x			x			x		x	x		x
		Synthetic Aperture		x	x	x		x			x				x	
		Orbital Debris	x		x	x	x	x		x	x		x	x	x	x
		Space Construction	x		x	x		x		x	x	x	x	x	x	x
	Exploration	Terrestrial Exploration		x	x	x	x	x	x	x	x		x	x		
		Orbital Exploration		x	x	x	x	x	x	x	x		x	x		
		Site Preparation	x		x	x		x		x	x		x	x	x	
		Asteroid Mining				x		x		x	x		x	x	x	

			Aggregation					Dispersion				Formation							Search			
			Gathering	Clustering	Flocking	Collab. Manip.	Overwhelm	Coverage	Exploration	Patrolling	Fractionated System	Relative Positioning	Connectivity	Patterns and Shapes	Network Building	Coordinated Motion	Self-Assembly	Structure Building	Blind Search	Source Search	Detection	Path Formation
Space	Constellations	Sensor Constellation						x		x		x	x	x	x	x				x	x	
		GPS Constellation						x		x		x	x	x	x	x						
		Communication Constellation						x		x		x	x	x	x	x						
		Power Transmission						x		x		x	x	x	x	x						
	Clusters	Fractionated Satellite									x	x	x			x						
		Synthetic Aperture						x				x	x		x						x	
		Orbital Debris	x	x		x						x	x	x		x			x	x	x	
		Space Construction	x	x		x						x	x	x		x	x	x	x	x	x	
	Exploration	Terrestrial Exploration						x	x	x	x	x	x		x	x			x	x	x	x
		Orbital Exploration						x	x	x		x	x			x			x		x	
		Site Preparation	x	x		x						x		x	x	x		x		x	x	
		Asteroid Mining										x	x	x		x	x		x	x	x	

Figure C.2.. Extended mapping of astronautical examples to the different types of *Tasks* and *Objectives* described in the Swarm Framework, including a mapping to specific application tasks.

D. CASE STUDY PARAMETER SWEEP DATA

Table D.1.: Raw data generated from the parameter sweep outlined in Chapter 5.

# of UAVs	0 Retasks	1 Retask	No Limit
2	6470	6395	6400
2	6780	6420	6380
2	6620	6375	6505
2	6855	6405	6420
2	6810	6455	6430
2	6800	6405	6460
2	6835	6390	6430
2	7035	6395	6425
2	6905	6410	6390
2	6900	6430	6420
5	3320	2645	2640
5	3310	2610	2610
5	3005	2610	2570
5	3250	2640	2600
5	3205	2600	2585
5	3300	2620	2585
5	3250	2630	2635
5	3540	2585	2640
5	3330	2650	2660

5	3265	2660	2610
# of UAVs	0 Retasks	1 Retask	No Limit
8	2300	1670	1640
8	2270	1670	1725
8	2385	1660	1670
8	2240	1655	1665
8	2340	1665	1660
8	2480	1665	1670
8	2410	1665	1675
8	2250	1705	1665
8	2310	1655	1655
8	2180	1665	1660
10	2005	1340	1355
10	1980	1335	1405
10	2090	1325	1405
10	45	1360	1390
10	85	1330	1330
10	60	1330	1400
10	40	1335	1350
10	30	1330	1355
10	05	1335	1345
10	55	1380	1325
12	05	1155	1165
12	85	1150	1145
12	05	1140	1150
12	65	1145	1140
12	25	1165	1160
12	50	1160	1160

12	50	1155	1095
# of UAVs	0 Retasks	1 Retask	No Limit
12	35	1175	1155
12	30	1155	1235
12	10	1160	1145
15	70	925	915
15	65	910	990
15	15	910	920
15	60	920	905
15	80	915	920
15	1505	985	910
15	1645	925	920
15	1515	915	975
15	1515	980	920
15	1600	930	915
20	1420	760	745
20	1595	755	675
20	1345	740	735
20	1575	740	725
20	1675	775	675
20	1345	745	685
20	1510	740	675
20	1575	730	675
20	1590	740	735
20	1420	745	695