The Swarm Garden: Human-Swarm Interaction for Self-Adaptive Art and Architecture

Jad Choaib Bendarkawi

April 2024

Advisor: Professor Radhika Nagpal

Submitted in partial fulfillment

of the requirements for the degree of

Bachelor of Science in Engineering

Department of Electrical and Computer Engineering

Princeton University

Collaboration Statement

The Swarm Garden project is a collaboration between the Self-Organizing Swarms and Robotics (SSR) Lab within the Mechanical and Aerospace Engineering Department and the Form Finding Lab within the Civil and Environmental Engineering Department. Contributors to this project include: Professor Radhika Nagpal, Professor Sigrid Adriaenssens, Dr. Merihan Alhafnawi, Dr. Lucia Stein-Montalvo, Victoria Chow, and Yenet Tafesse. Yenet Tafesse was a key contributor to the development of the human-swarm system described in this thesis, especially on the front of developing a Wearable Device for Human-Swarm Collaborative Dance Performance and Improvisation.

I hereby declare that this Independent Work report represents my own work in accordance with University regulations.

Jad Bendarkawi

Jad Choaib Bendarkawi

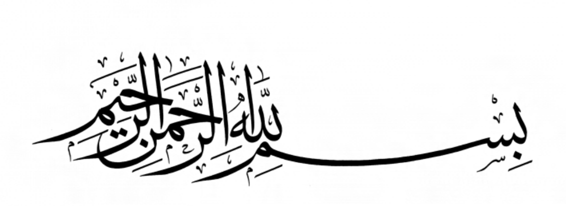
The Swarm Garden: Human-Swarm Interaction for Self-Adaptive Art and Architecture

Jad Choaib Bendarkawi

Abstract

As part of the Self-Organizing Swarms and Robotics Lab's project in collaboration with the Form Finding Lab titled, The Swarm Garden, this thesis describes the design, implementation, and evaluation of human-swarm interactions to create "living-like" architectures that invoke nature to make people feel a sense of health and well-being. At the intersection of swarm intelligence, architectural design, art-making, and dance, The Swarm Garden seeks to demonstrate an experimental, nature-inspired interactive architecture exhibit where flower modules bloom in response to human presence and can exhibit complex long-range and real-time responses through self-organization. Through the investigation of proximity and vision based interaction modalities for resource constrained devices, applications of expressive and embodied swarms, and development of a robust, canvas-like human-swarm system, we explore the potential and realization of architectural swarms in human spaces. The project culminates in a public exhibit titled *The Swarm Garden: An Interactive Architecture Exhibition*, held at the Princeton University Lewis Center for the Arts CoLab, where multiple visitors constantly create new outcomes through interaction and stochasticity so that each visit is a new experience. This work contributes to the ongoing research of the applications of swarm intelligence to art, architecture, and human expression. Ultimately, research in this area can pave the way for the design of human spaces that lead to better health outcomes, human-robot interaction that improves our quality of life and well-being, and beautiful experiences in art and art-making through emergent interaction outcomes. The Swarm Garden also serves as a proof of concept of the combination of swarm intelligence and buckling sheet technology from the Form Finding Lab that will be the launchpad for larger scale, self-adaptive facade architecture applications.

Acknowledgements



I want to express my gratitude to Professor Radhika Nagpal for her unwavering support, commitment to equity and diversity, inspiration and creativity, and offering me the opportunity to take part in this amazing project that intersects art and robotics. The chance to work on an engineering project for the sake of finding beauty and human well-being is very rare and meaningful to me, and I'm so honored to be a part of this work. I also would like to thank Dr. Merihan Alhafnawi for being a great mentor for me throughout this project and for always being available to help! Thank you to Yenet Tafesse for being an awesome undergrad collaborator and friend throughout this project. Through many late nights building modules, many coding / brainstorm sessions, and many trips to Small World, I couldn't have done it without you! Thank you to all of the people at the SSR Lab who have encouraged me throughout this process and have played a role in fabricating The Swarm Garden. A big thank you to our collaborators from the Form Finding Lab – Professor Sigrid Adriaenssens, Dr. Lucia Stein-Montalvo, and Victoria Chow for the 3D renderings / animations, architectural expertise, and collaboration to bring The Swarm Garden to life. Thank you to Azariah Jones for joining our team and sharing her art with us through her dance performance alongside The Swarm Garden! Thank you to Baffour Osei for being an amazing Lab Manager in the F-Wing and Maureen Novozinsky and Patrick Richichi from OIT for helping us a ton with sorting out our network / internet access. Thank you to Professor Yasaman Ghasempour for being my second reader. Thank you to the ECE Department and SEAS for providing generous funding and to Lori Bailey for being an amazing administrator and advocate for my success. A big big thank you to all the friends and family who have supported me throughout this year and all who attended The Swarm Garden exhibition!

I dedicate this thesis to my parents, Youssef Bendarkawi and Cherifa Lousa.

Contents

Chapter 1

Introduction

1.1 Background and Motivation

Today, current human-designed architectures are for the most part static, designed with rigid and non-reusable materials that remain fixed even as the environment and occupant-needs change. In contrast, living architectures, from plants to cells to bee hives, are constantly evolving in response to their environments with both immediate and long-term adaptations. This capability emerges from self-organization, whereby individual agents interact locally with each other and sense their local environment, and through the networked interactions, create globally complex responses. The field of swarm intelligence investigates the algorithms and technology inspired by such collectives for the design of novel engineered systems. The Swarm Garden aims to bring concepts from swarm intelligence into the design of human spaces and self-adaptive architecture.

The potential of architectural swarms is particularly powerful in that it circulates around providing a novel lens to apply human-swarm interaction as well as achieving happiness through architecture by utilizing adaptive structures to respond to human needs. In this way we apply swarm robotics to augment human well-being by making autonomous decisions that positively alter human spaces i.e autonomously allowing sunlight to enter a room by self-adapting to human and environmental factors. Combining HSI and architecture, we can push the boundary of current self-adaptive architectures that utilize more materials focused [1][2] than human-centered approaches to become more attuned and responsive in supporting human happiness and well-being. We can look to previous applications of HSI in art-making, human expression, dance, etc. to guide such human-centered approaches [3][4][5][6]. Through this work we also find an opportunity to explore new and innovative methods for expressive swarms as tools for dance and performance art. By investigating the intersection of HSI and dance, this application creates a novel method for emergent, synergistic, and beautiful improvisational and choreographic performance outcomes, re-imagining performance environments as human-swarm collaborative and self-adaptive. We envision futures where dancers, artists, and performers can utilize architectural swarms like The Swarm Garden as extensions of their artistic works and utilize swarm intelligence to create embodied experiences with technology. Moreover, on the motivation for this thesis, as we are building an expressive, architectural swarm exhibit for beauty and human happiness and well-being, there is an exciting intersection to explore around the role of robots in society and how this exploratory human-swarm interactive experience can serve as an opportunity to gauge and shift human sentiment on human-swarm interaction and human-robot collaboration at large.

As we enter an age of human society where robots and autonomous systems are becoming more frequent in our daily lives, we typically attribute a necessary utility or function to these machines. At the same time, as our technologies evolve to become more complex and powerful, there is a fear that robots will replace humans, or some day take over the world as we've seen in speculative fiction and popular science fiction media. In this way, we delineate a separation between the human and robot experience, rather than seeing human-robot interactions as opportunities to collaborate and coexist for the betterment of health and well-being. Technologists and artists like Anicka Yi, who question whether machines could relate to us in a more holistic way, encourage humans to embrace our position as symbiotic living entities and design machines and technology that reflect this. Why do our technologies instill so much fear in us? Why do our lives today feel so alienating when our technologies are supposed to improve our well-being? Why do we feel so disconnected when our inventions are meant to connect us? What if machines were more like animals and plants? Questions like these motivate Anicka Yi's piece, *In Love with the World*, a Hyundai Commission open exhibit at Tate Modern that imagines a 'natural history of machines' by allowing various bio-inspired aerial robots to autonomously roam and interact with human visitors, creating novel and holistic human-robot interactions [7][8].

The Swarm Garden attempts to similarly address these questions that intertwine human and machine, offering an opportunity for humans and robots to create unique, holistic experiences through human-swarm interaction and self-organization. Both human and machine work together to create beautiful outcomes that improve happiness and health, leveraging nature-inspired design as a substrate to connect our experiences as one. The field of human-swarm interaction provides us the technological groundwork to create opportunities for us to reimagine our relationship with technology, and through nature-inspired design and swarm intelligence applied to an architectural exhibit, The Swarm Garden serves as a beacon for us to speculate a joyous future of coexistence between humans, machines, and nature.

Figure 1.1: A photo of "aerobes" from *In Love with the World* a Hyundai Commission open exhibit at Tate Modern [8]

1.2 Objective and Project Scope

At the intersection of swarm intelligence, architectural design, and art-making, The Swarm Garden seeks to demonstrate an experimental, nature-inspired interactive architecture exhibit comprised of 36 robotic flower modules that interact with people through self-organization and human-swarm interaction. This thesis mainly focuses on the ambient and stochastic experience of visitors, centering on how users naturally interface with exploratory interaction modalities to create unique experiences and outcomes. We also detail one pre-programmed mode and one specific task-based interaction. We provide the control methods and interaction modalities to interface with The Swarm Garden, but ultimately leave goal definition up to each individual operator, allowing visitors to explore and become acquainted with the swarm themselves, discovering emergent behaviors to create novel and beautiful experiences along the way.

Figure 1.2: *The Swarm Garden: An Interactive Architecture Exhibition* (taken by Lori Nichols).

This thesis will review my contributions to the development of the human-swarm system built to deploy The Swarm Garden, including the overarching human-swarm system design, inter-module communication and localization system, and several interaction modalities. The Swarm Garden system is intended to be scalable, responsive, and true to the components of human-swarm systems similar to projects this one is inspired by – I especially take care in defining conventions and a system that can easily integrate new interaction modalities to enable future work built on the platform. In this way The Swarm Garden can serve as a test-bed for different interaction modalities for expressive swarms across different architectural, performance art, and general art-making applications. We will observe a case of integrating a wearable device with the base swarm system by touching on the work of Yenet Tafesse, a collaborator working on The Swarm Garden's wearable device for human-swarm collaborative dance performance and improvisation.

After orienting to the hardware and overarching system design, I will focus on the software development of methods for inter-module communication and human-swarm interaction modalities along with design considerations specific to human-swarm systems (i.e input timing, visual feedback, state estimation and resolution, etc.). I will also review the capabilities and limitations of the Arduino Nicla Vision, a recently released resource-constrained Arduino camera module, as they relate to the attempted and successful interaction modalities developed. I particularly focus on proximity and vision-based interaction modalities, and likewise address the constraints of resource-constrained devices for computer vision techniques used. Throughout this thesis I will aim to address single points of failure and centralizing elements that are either remediated or deferred to future work in order to prioritize keeping agents simple and decentralized as best as possible given developmental challenges and time constraints of the exhibition. Finally, I describe the public exhibition of The Swarm Garden and discuss the design and results a user study assessing the larger audience sentiment as well as an evaluation of the HSI interface with respect to a human-swarm collaborative improvisational dance performance.

1.3 Literature Review

1.3.1 Human-Swarm Interaction: Components & Advantages

Before delving into applications of swarm intelligence and human-swarm systems, we begin broadly by examining key components (Fig. 1.3) and advantages of these systems to provide the language and context behind the development of The Swarm Garden.

Robotic swarms consist of multiple robots that coordinate autonomously via local control laws based on the robot's current state and nearby environment, including neighboring robots. In the development of robotic swarms, methods for identifying and assigning neighborhoods, conventions for inter-robot communication and message propagation, and an input-output system for human-swarm interactions are encompassed. Swarm systems are particularly advantageous in their robustness to failure of individual robots and scalability due to the simple and distributed nature of their coordination. This centers a design principle of human-swarm systems around avoiding single points of failure and an emphasis on decentralization which we will attempt to address and make note to address in the current and future development of The Swarm Garden. [9]

Furthermore, an important issue for the supervision of robotic swarms is the ability to efficiently convey information about the swarm's current state, its possible future states, and the effects of human input on its behavior. From the perspective of the operator, we recognize the constraints the swarm interface poses for means of communication and thus rely on methods for state estimation, visualization, and control to facilitate interactions between humans and the swarm [9]. State estimation by the operator typically involves a one or a combination of swarm-level visual, auditory, or tactile feedback and can be aided by supplementary visualization tools i.e 3-D visualization tools [9][10]. This component is especially important to ensuring operator understanding of interaction outcomes, informing the timing of control inputs, and improving overall intuitiveness of the human-swarm interface. Due to various hardware constraints within The Swarm Garden's development, we strongly consider the impact of design decisions on the ability for operators to retain swarm-level context and prioritize human input to visual feedback connection. Honing in on this connection, we also consider a key component surrounding the cognitive complexity and delay of human-swarm interactions. Reducing delay between interaction input to visual output from the swarm is especially crucial in reducing the cognitive complexity and effort of the human operator required to control multi-robot systems. The Swarm Garden also benefits cognitively from being a static structure (i.e agents are not freely moving in 3-D space) and acts as a canvas-like interface which reduces overall complexity that would come with building interactions with multi-agent motion planning and coordination. This use of the "canvas" serves as a familiar point of reference for operator intuition to be built around expected interaction outcomes, thus facilitates successful interactions [6].

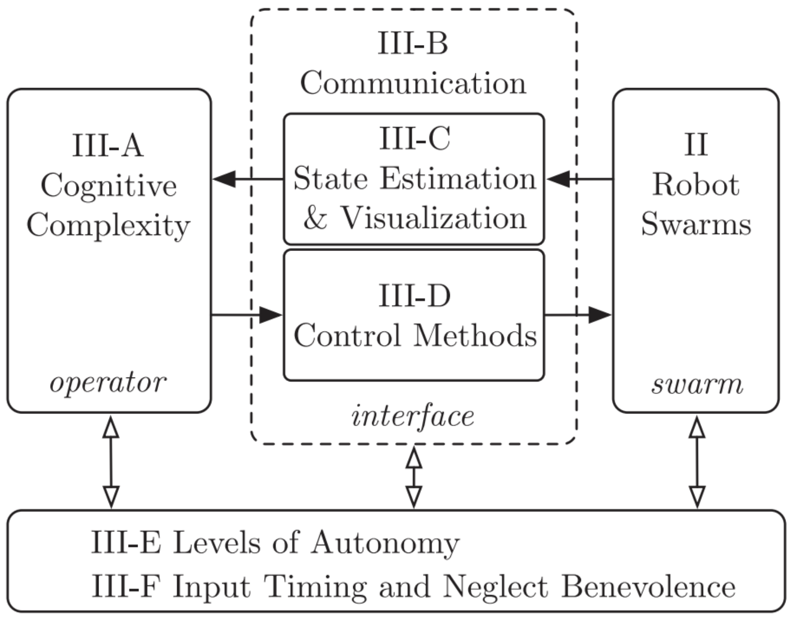


Figure 1.3: Key components of a human–swarm system, with an operator solving complex tasks and communicating with a swarm through an interface to receive state feedback and send inputs using appropriate control methods [9].

1.3.2 Applications in Art, Architecture, & Expression

We will now discuss the application and implications of human-swarm systems in art, architecture, and human expression as motivation and inspiration for the interaction modalities explored and implemented in The Swarm Garden.

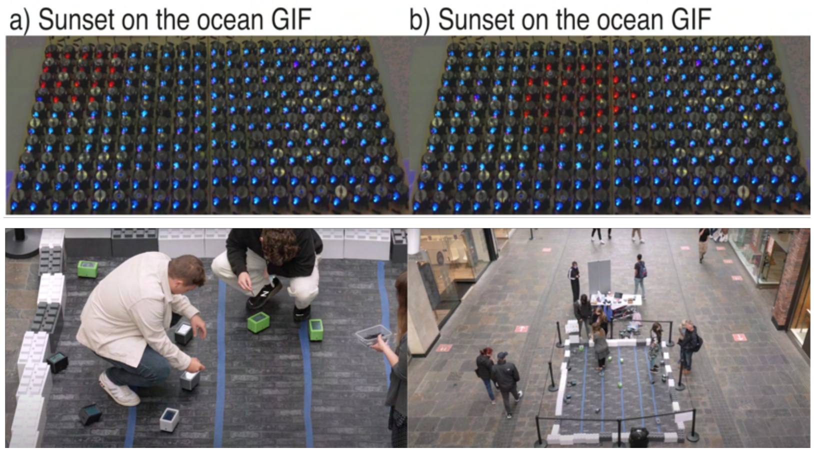


Figure 1.4: Video projection on the Robotic Canvas human-swarm system [6] (top row). Opinion-mixing live event via MOSAIX human-swarm system [3] (bottom row).

Expressive Swarms

The Swarm Garden is inspired by several works of *expressive swarms*, or the use of robotic swarms to convey and evoke abstract ideas, emotions, images, and creative expressions. Works like Robotic Canvas (Fig. 1.4) and other "painting" inspired multi-robot interactive systems (Fig. 1.5) are particularly relevant in that swarms are used to display projected images and utilize gestures and kinetic inputs to physically paint onto the swarm / swarm's environment [11][6]. Art-making in this way becomes a human-swarm collaborative performance between artists, digital content, and robotic agents. The work of Robotic Canvas is particularly relevant in its setup as a static (non-moving) swarm, where the agents act as pixels and respond via LEDs, similarly to the approach we utilize in The Swarm Garden. We also look to works like MOSAIX where robotic swarms can be embedded in social settings, allowing crowds of humans and robots to interact to create productive and collaborative social interaction outcomes. For the purposes of MOSAIX, humans can interact with individual touch screen agents to submit opinions to a larger sea of opinions, allowing users to observe group decision making outcomes in the physical world (Fig. 1.4) [3]. Materializing opinions and attitudes in this way allows humans to extract meaning from the swarm about consensus, aggregate group opinion information, and participant engagement in a "deliberative democracy" setting [12]. The swarm thereby acts as a tool to facilitate human social interactions, including opinion mixing, brainstorming and networking, and reveals ways in which human collectives can reach mutual understanding, productive collaboration, and new forms of creative expression [3].

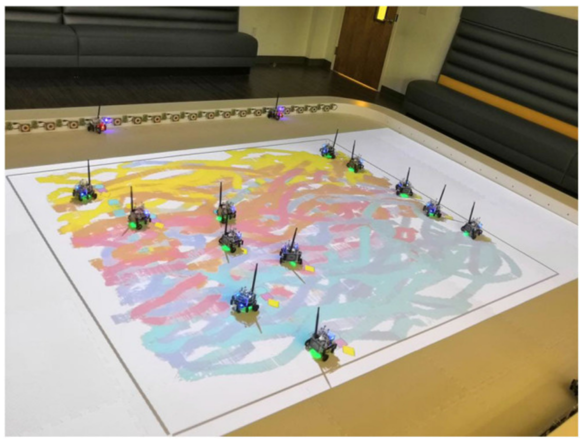


Figure 1.5: A group of robots generate a painting based on the densities specified by a human user for five different colors. The robots lay colored trails as they move throughout the canvas. The painting arises as a result of the motion trails integrating over time [11].

Embodied Swarms

The Swarm Garden also draws on inspiration from explorations of *embodied swarms*, or the use of robotic swarms as extensions of the physical human body to realize intuitive interactions within the environment. A key component of embodiment is the sense of body ownership and agency, and through a synchronized control approach where swarm robots match positions to the user's position in the remote physical world, it is possible to achieve such sentiments in human operators [13]. Drawing inspiration from this framing is especially important to building a system that has control intuition where operators can engage and interact with the swarm in ways that feel natural and familiar as done in motion-mimetic systems like Keepon Robots (Fig. 1.6) and a gesture-based human-multi-robot interactive display (Fig. 1.7) [14][15]. This is especially relevant to our development of The Swarm Garden as a tool for human-swarm collaborative dance performance and improvisation as well as generally to inform and motivate the interaction design of successful and intuitive gesture based modalities. Successful intersections between swarm robotics and dance have also been shown to root the design of gestural, expressive commands in the designation of body movement and their emotional associations from dancers and choreographers themselves, categorizing body motions into perceived emotions and using these to inform expressive commands sent to the swarm [5].

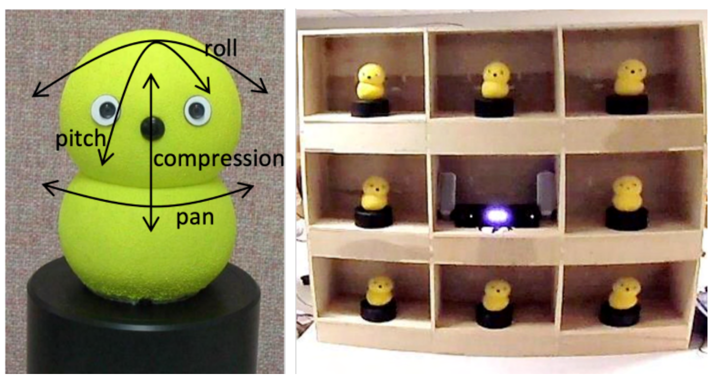


Figure 1.6: Diagram of the axes to which gestural dance commands are translated (left). 3x3 grid of 8 dancing Keepon robots with Kinect sensor in the middle (right) [14].

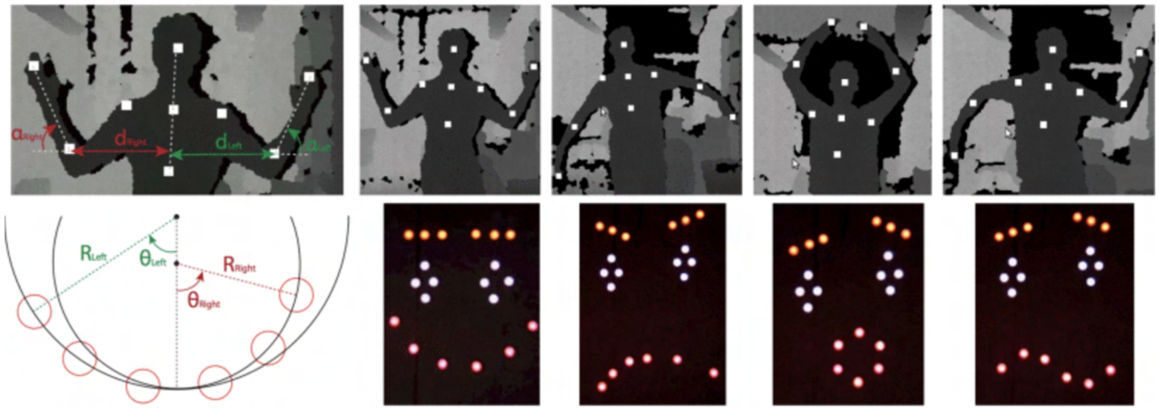


Figure 1.7: Gesture based constrained-shape morphing for multi-robot swarm. A Kinect camera extracts human pose data generating keypoints to map gestures into swarm-level expressions [15].

Architectural Swarms & Nature-Inspired Design

In the realm of architecture where robotic swarms have been used primarily for fabrication [16], swarm-systems as installed structural components of the human built environment are still understudied and underdeveloped. A company called HelioTrace, shows the potential use case for robotic facades for self-adaptive lighting and climate control in buildings (Fig. 1.8) [17]. The Swarm Garden in the architectural context aims to explore this gap, where we aim to explore the power of self-organization and swarm intelligence to create large-scale, long-range responses in facade designs, that optimize for particular architectural outcomes: shading vs. artificial lighting, views vs. privacy, solar gain vs. overheating, and daylight vs. glare [18]. We also draw on living architectures and biofacades (Fig. 1.8) from a design perspective in our goal to achieve an aesthetically and functionally nature-inspired system that can live in harmony within the human built environment and mimic the qualities of self-organizing structures found in nature.

Figure 1.8: Rendering of HelioTrace's self-adaptive, robotic architectural facade [17] (left). Biofacade produced with microalgae to increase air quality indoors [19] (right).

Finally, as we approach this 'natural' quality in our design of The Swarm Garden, projects like MIT CSAIL's Distributed Robot Garden (Fig. 1.9), where they developed an aesthetically pleasing educational platform that can visualize computer science concepts [20], serve as a guidepost for designing at the intersection of nature, swarm robotics, and art in ways that compliment desired interaction and sentiment outcomes among user groups.

Figure 1.9: Photo of MIT CSAIL's Distributed Robot Garden education platform for computer science fundamentals and inspiring young students to pursue programming and robotics [20].

Chapter 2

Design and Implementation

2.1 Swarm System Design

In this section we will discuss various swarm configurations of The Swarm Garden, the hardware components of the flower modules, the capabilities and limitations of the main processing and sensing unit of the modules (Arduino Nicla Vision), and examine hardware design decisions and constraints as they relate to the development of successful human-swarm interactions.

2.1.1 Swarm Configurations

The Swarm Garden is an interactive architectural swarm display comprised of 36 modules arranged in 3x3 grids. Though initially designed to be mounted on freely movable wheeled stands, for the purposes of the exhibit, we use freely movable podiums of various heights to create a staggered mosaic of flowers (Fig. 2.1). This less rigid shape achieves a more nature-inspired / organic aesthetic, however creates interesting problems for localization as we will discuss in Section 2.2.3. This modular configuration approach not only allows us to create the flower garden wall, but also be able to easily try out different configurations, allowing us to observe how various configurations correlate to interaction outcomes. This will be especially useful in observing how the swarm might behave in future, larger scale self-adaptive architecture applications. What if we were completely surrounded by multiple walls of The Swarm Garden? How do different shapes of the swarm correlate to different interaction outcomes and user engagement? These questions can help guide future configuration exploration beyond the staggered exhibition setup.

Figure 2.1: Original configuration with 36 modules arranged in square 3x3 grids (top) and final configuration with 36 modules arranged using staggered podiums (bottom). (Rendering by Victoria Chow)

While we are designing The Swarm Garden for the purposes of a standalone architecture exhibit in a controlled environment, we ultimately envision this technology to be deployed in human spaces like offices, lobbies, and outdoor spaces, where the benefits (i.e adaptive natural lighting and shading) and beauty of self-adaptive swarms become embedded in the human built environment. Some of these future renderings include a window mounted swarm garden that allows various levels and patterns of natural light and shade, outdoor pavilion enclosed in a swarm garden ceiling, and swarm garden ceiling for indoor spaces with skylight windows (Fig. 2.2).

Figure 2.2: Future applications of The Swarm Garden as a self-adaptive facade for lighting and shading. (Renderings by Victoria Chow)

2.1.2 Arduino Nicla Vision

The Arduino Nicla Vision is a compact yet powerful microprocessor designed for vision-based applications. It's equipped with a 2MP color camera, 6-axis motion sensor (IMU), integrated microphone, infrared / time-of-flight proximity sensor, and wireless communication capabilities, making it suitable for projects that require image processing and interaction with the surrounding environment (Fig. 2.3). In the context of The Swarm Garden, each flower module relies on the Arduino Nicla Vision as its main processing unit to interpret stimuli and facilitate communication within the swarm. We utilize OpenMV to run MicroPython on the module and interface with the camera, proximity sensor, Wi-Fi capabilities, and on-board LED.

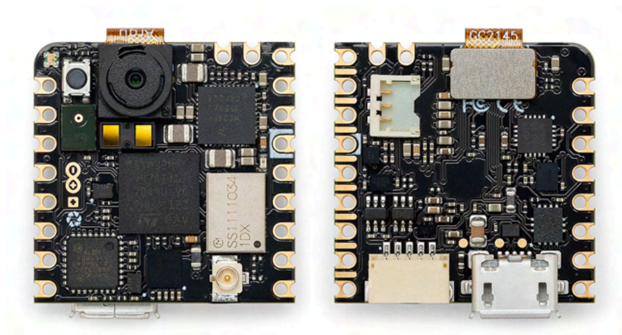


Figure 2.3: Arduino Nicla Vision module, front (left) and back(right) [21].

Throughout our use of the Arduino Nicla Vision we found issues particularly on the fronts of storage / memory, programmability, and proximity sensing in relation to developing software for an entire swarm. Due to the resource-constrained nature of the Nicla, machine learning models for object detection and gesture recognition were difficult to run directly on the Nicla as compressing the models would sacrifice accuracy and the space they took up would leave little room for our main module interaction logic to live. We discuss this in more depth in Section 2.3.2 as we explore methods for offloading computation to a Central PC and streaming data off the Nicla remotely. Another challenge was the issue of reprogrammability and debugging with OpenMV/Micropython. Given our choice of IDE, while flashing and programming Niclas could get up and running quite quickly for a single Nicla, we were unable to take advantage of technologies like WebREPL or Arduino Over-the-Air (OTA) that upload programs wirelessly, allowing robotic swarm agents to receive new code without having to plug in and test every individual module, which would significantly reduce debug time. Finally, in our use of the on-board proximity sensor we found that when there were objects in front of the sensor but outside the maximum detectable range, the values read would spike between large and small values, causing interactions to be less accurate (i.e randomly triggering state changes when no operators are present in front of the swarm). To circumvent this, we attempt to keep a moving average of sensor readings, however this does not fully remove the problem for sustained, consecutive readings of an environment with no operators in the proximity sensor range. A wall parallel and centered to The Swarm Garden within the proximity sensor's maximum range would be needed to fully remediate this issue.

2.1.3 Flower Module Design

Each flower module contains a minimalist shimstock sheet element that can spontaneously and efficiently fold by exploiting the bistability of confinement [22] – essentially the ability for the sheet to buckle into a flower-like shape when pulled through a ring. Using a long screw attached to a stepper motor (Fig. 2.4), a simple mechanism pulls the sheet through a ring, causing the sheet to buckle into various floral patterns depending on the size of the sheet, distance pulled through the ring, and number of sheets. Sheets of varying thickness and pliability achieve different buckling patterns, allowing us to achieve a diverse and beautiful array of flowers with varying numbers of lobes as one would observe in a garden of flowers in nature (Fig. 2.5) [22]. A time-of-flight sensor is attached to the long screw, allowing us to gauge the distance the sheet is pulled through the ring. We gate the blooming behavior using this sensor to prevent collisions within the module with the supporting electronics (PCB, battery pack, etc.) as well as preventing the Arduino Nicla board from pressing against the front panel or falling out. As this sensor is pointing backwards at the panel of electronics, in testing, we would obtain inaccurate proximity readings given the irregular topology of the back panel, which would cause premature stopping of the blooming mechanism. We utilize a flat, 3D printed surface to ensure consistent proximity readings to ensure full blooming behaviors occur.

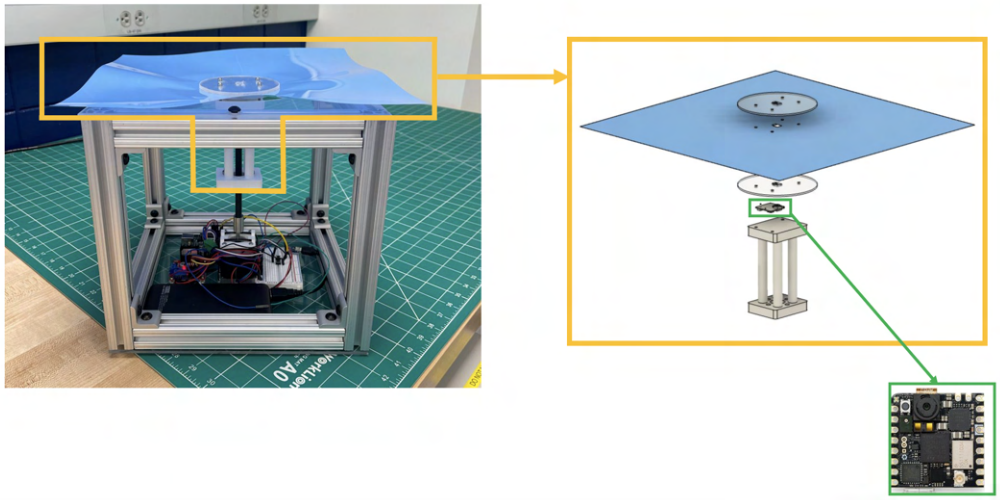


Figure 2.4: Placement of the Arduino Nicla Vision in the center of the module and long screw with stepper motor mechanism connected to the sheet element.

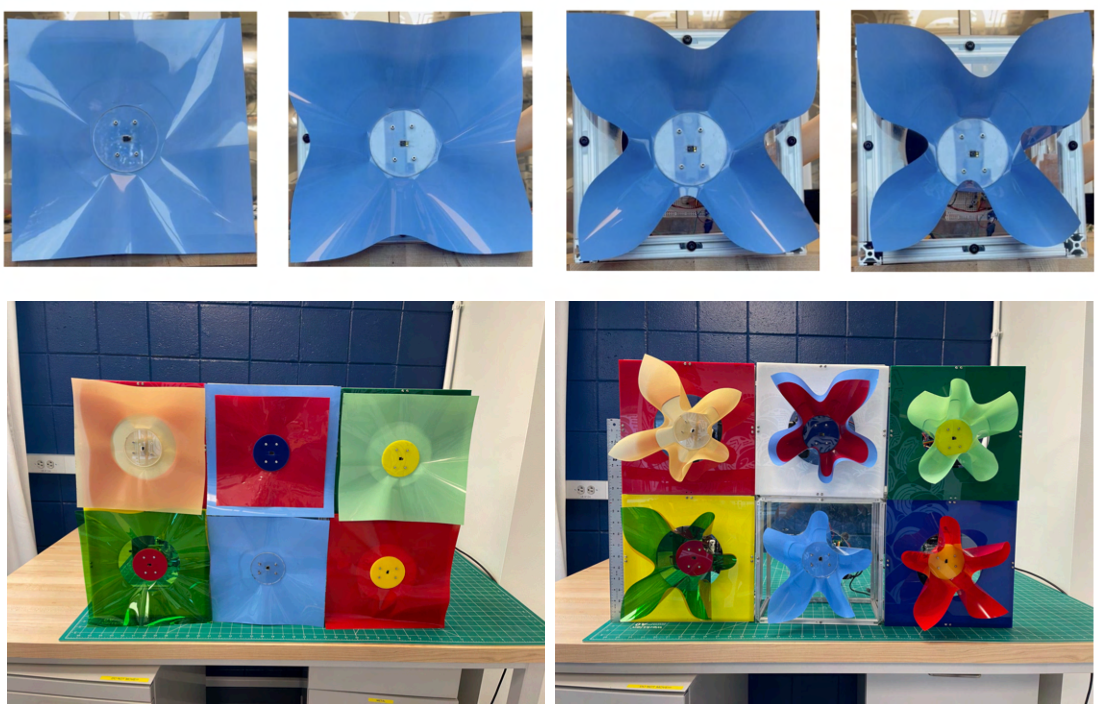


Figure 2.5: The pulling procedure of the plastic sheet through the hole created in the acrylic front (top row). The pulling procedure on a 2x3 grid (bottom row).

This mechanism is encased in an aluminum base frame, where the electronics live on the back panel and the front panel is a piece of frosted acrylic with hole cut out for the sheet element to be pulled through. The frosted acrylic is used to diffuse the light of a LED NeoPixel strip running along the perimeter of the back panel. In this way we enable another simple and effective form of visual feedback and state estimation as different colored light can serve as a response to different stimuli. Providing swarm level readout visual feedback is important to creating a closed-loop interaction by enabling the human to confirm that the swarm has understood the intended interaction, and using LED's to further drive operator understanding reduces cognitive complexity, making the human-swarm interaction more likely to succeed [9][6]. The frosted acrylic also helps hide some of the electronics within the module to create a more seamless visual experience for exhibit visitors, allowing them to focus completely on the human-swarm interaction experience. The main components of the hardware design discussed in this section can be found in Figure 2.6.

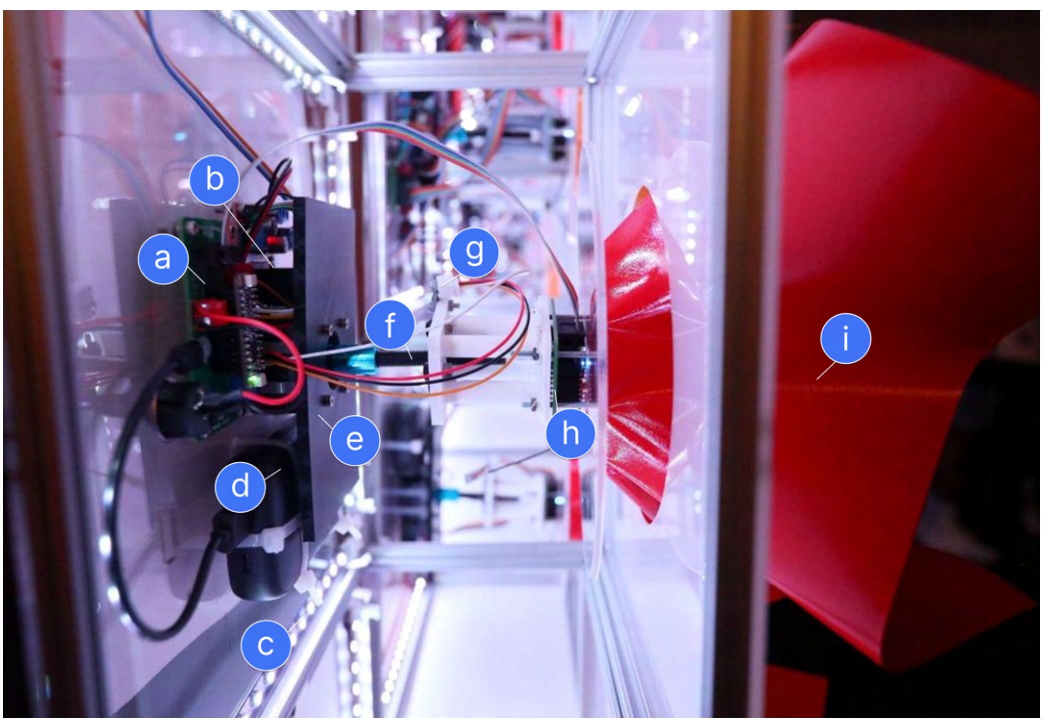


Figure 2.6: Main components of the flower module. a) Expander board with MCP23017. b) Motor driver and stepper motor. c) LED NeoPixel Strip. d) Rechargeable battery pack. e) 3D printed flat surface. f) Long screw with coupler. g) Time-of-flight sensor for bloom thresholding. h) Arduino Nicla board. i) Shimstock sheet.

In initial testing, we found the speed of the motor to be relatively slow with the time to fully bloom from flat ranging from 30 to 45 seconds. This is the fastest rate for the particular motor used with the load weight of the long screw and attached Nicla board, shimstock sheet, and supports. This is significantly slower than the rate of reaction of the LED strip which is perceptually simultaneous to human operators. These varying visual feedback speeds raise interesting considerations for cognitive delay and inter-module communication that will be further addressed in Section 2.2.6.

2.2 Inter-module Communication

In this section we will discuss the inter-module communication system built to support local interactions between the environment and neighborhoods of swarm agents. Through manipulating module state variables, neighbor identification and localization, and defining message content and a robust propagation scheme we successfully exchange messages between neighbors, Central PC, and localization camera through the swarm network.

2.2.1 Module State Variables & Network Settings

Each module keeps a unique `module_ID` that is assigned based on the ID of a unique AprilTag from family 36H11 that is attached to the back of the module (there are no duplicate tags within the swarm). A dynamic list of neighbors is also kept to keep track of the nearest neighbors, along with their relative position for more discrete directional message propagation (how this list is maintained and updated is explained in Section 2.2.3). The current `mode` is also maintained in order to switch between interaction modalities, along with local data about the current state of the module (Fig. 2.7): `LEDColor` for the on-board LED Color, `LEDStripColor` for the

LED NeoPixel strip color diffused by the frosted acrylic front, and a check for each module's time-of-flight sensor's blooming and flattening thresholds to gate blooming to hardware-safe values (`bloomThresh` and `unbloomThresh`) for updating the current state of how far the sheet element is pulled through the ring. Each module also statically stores its corresponding shimstock sheet color (i.e red, orange, or yellow) for later use in interaction modalities. Through inter-module communication as well as communication with the localization camera and Central PC, the dynamic state variables evolve and transform to give visual feedback in response to interactions.

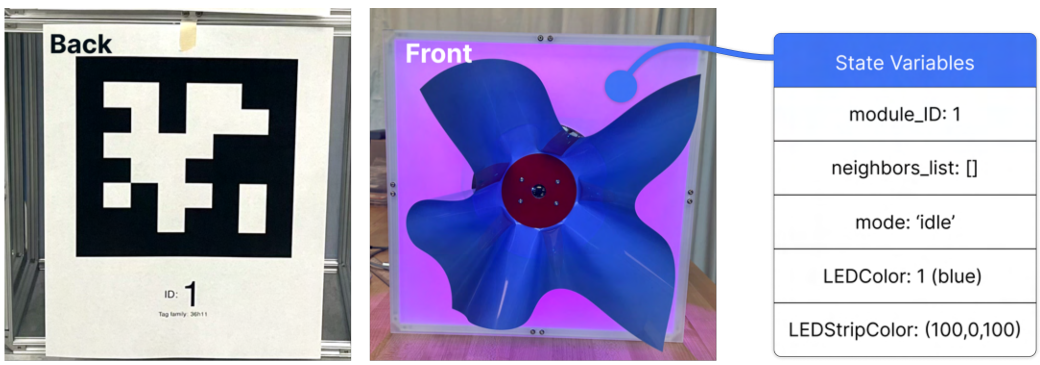


Figure 2.7: Relevant state variables for a single module visualized.

There are also module specific network settings that allow for successful communication and vision-based interactions to occur across the swarm. We assign a unique listening port for messages and unique IP and web port for streaming live camera data to the Swarm Vision Visualizer web interface (explained further in Section 2.3.2).

2.2.2 Swarm Network Topology

The network used to deploy The Swarm Garden (Fig. 2.8) utilizes Wi-Fi for routing the listening ports of the modules. We also connect our Wi-Fi network to the internet for running the Swarm Vision Visualizer with pose estimation as it imports the necessary machine learning packages live as will be discussed in Section 2.3.2.

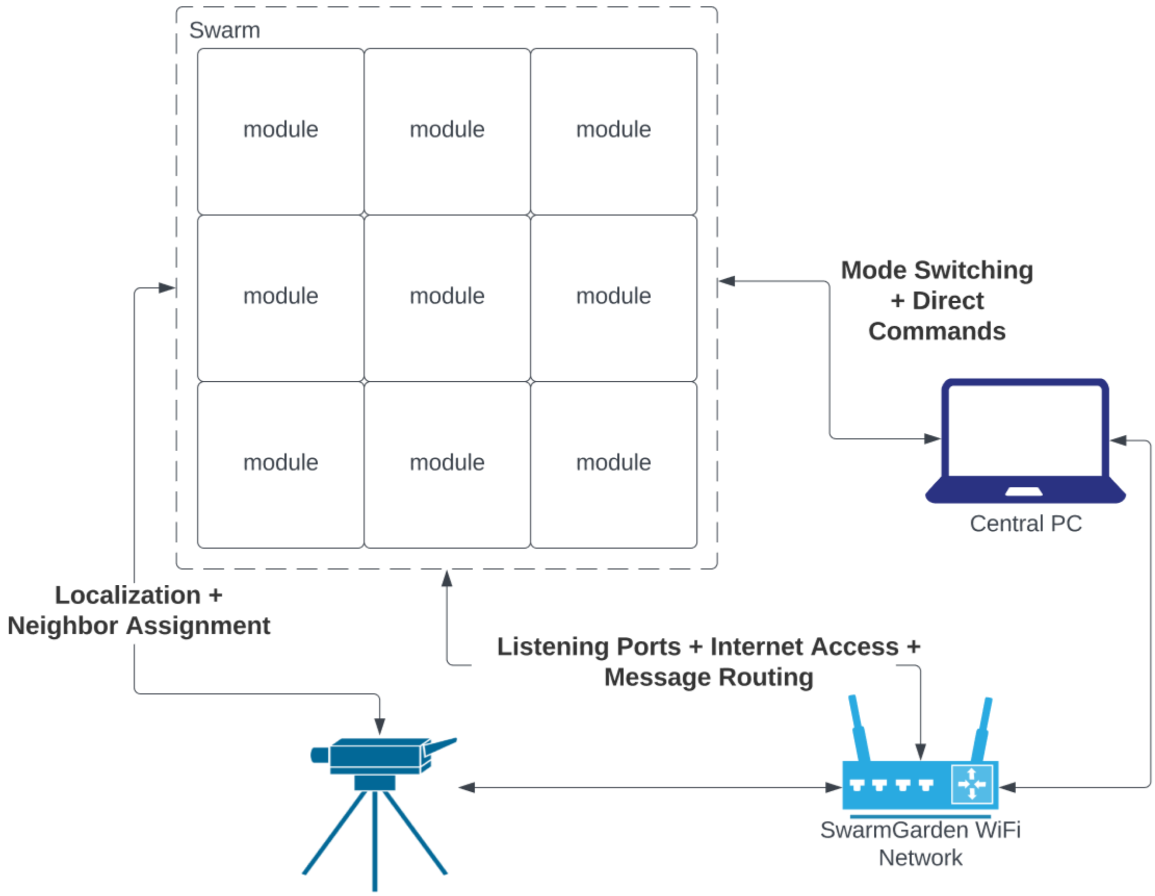


Figure 2.8: Swarm network topology including the main components: Central PC, Localization camera, and Swarm relative to the router's Wi-Fi network.

2.2.3 Neighbor Identification & Localization

Each module has a unique AprilTag from the 36H11 tag family attached to its back, denoted as `module_ID`. AprilTags are a type of two-dimensional bar code designed to encode small data payloads, allowing them to be detected from long ranges. They are also designed to perform with high localization accuracy, allowing precise 3D position of AprilTags to be calculated with respect to the camera [23]. The localization camera pointing at the back of the swarm (Fig. 2.9 and Fig. 2.11) captures the x and y positions of these tags, facilitating the identification of the nearest n (default $n = 8$) neighboring tags by Euclidean distance. These neighbors are assigned relative positions (`top, topright, topleft, bottom, bottomright, bottomleft, right, left`) and are kept with their corresponding `module_ID` as tuples in the format `(relativePosition, module_ID)`. Non-adjacent neighbors are marked as `'far'` to limit direction-specific (i.e propagating an LED color towards the bottom left) commands to adjacent modules. The `neighborsUpdate` command is sent continuously from the localization camera to ensure that each module dynamically updates its list of neighbors based on any configuration changes, fostering an agile and responsive swarm system. In this way, we avoid single points of failure from singular module outages as the swarm will simply continue to update neighbor assignments across all modules, thus always retaining swarm-level connectivity. Algorithm 1 and Algorithm 2 provide the logic for the main loop sending out neighbor assignments to the swarm as well as the helper functions developed to calculate relative positions as described in this section.

Algorithm 1: Neighbor Identification: Main Loop

```
Function main():
    while True do
        clock.tick();
        img ← sensor.snapshot();
        detected_modules ← find apriltags in img with parameters
          fx = f_x, fy = f_y, cx = c_x, cy = c_y;
        for tag in detected_modules do
            neighbors ← main(calculate_neighbors(tag, detected_modules,
              num_neighbors=6));
            sendData ← "neighborsUpdate" +tag.id()+
              main(neighbors_string(tag.cx(), tag.cy(), neighbors));
```

Algorithm 2: Neighbor Identification: Helper Functions

Function `calculate_distance`(*tag1, tag2*):
  return $\sqrt{(tag1.cx() - tag2.cx())^2 + (tag1.cy() - tag2.cy())^2}$;

Function `calculate_neighbors`(*tag, tags, num_neighbors*):
  $neighbors \leftarrow []$;
  $sorted\_tags \leftarrow$ sort tags based on distance to the current tag;
  for $other\_tag$ ***in*** $sorted\_tags[: num\_neighbors]$ **do**
    if $tag \neq other\_tag$ **then**
      append $other\_tag$ to $neighbors$;
  return $neighbors$;

Function `relative_position`(*tagCX, tagCY, neighborCX, neighborCY*):
  $dx \leftarrow tagCX - neighborCX$;
  $dy \leftarrow tagCY - neighborCY$;
  if $dx > 80$ ***and*** $dy > 100$ ***or*** $dy > 112$ ***or*** $dx > 112$ **then**
    return *"far"*;
  else if $tagCY < neighborCY$ ***and*** $dy < 112$ ***and*** $dx < 10$ **then**
    return *'bottom'*;
  else if $tagCY > neighborCY$ ***and*** $dy < 112$ ***and*** $dx < 10$ **then**
    return *'top'*;
  else if $dx < 0$ ***and*** $tagCY < neighborCY$ **then**
    return *'bottomright'*;
  else if $dx < 0$ ***and*** $tagCY > neighborCY$ **then**
    return *'topright'*;
  else if $dx > 0$ ***and*** $tagCY < neighborCY$ **then**
    return *'bottomleft'*;
  else if $dx > 0$ ***and*** $tagCY > neighborCY$ **then**
    return *'topleft'*;

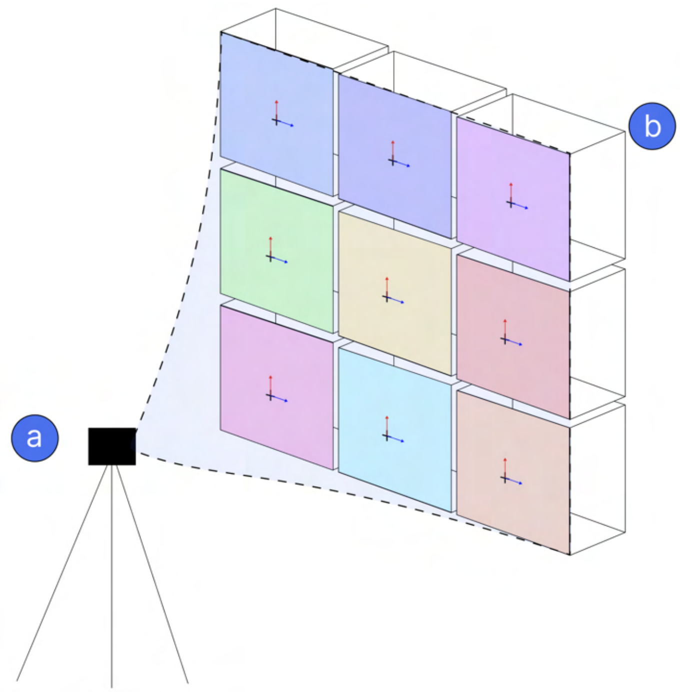


Figure 2.9: Localization setup facing the back of the swarm. a) OpenMV Camera used to capture AprilTag positions, perform neighbor assignment, and broadcast `neighborsUpdate` messages to the swarm. b) 3x3 swarm configuration where each color represents a unique AprilTag attached to the back of each module.

We must also consider the potential drawbacks of using a singular camera to perform localization rather than a hardware-based determinant of neighbors. For example, Kilobots used in Robotic Canvas and other swarm-based projects, utilize infrared pointing to the ground surface they are placed on, allowing only physically nearby agents to receive messages as they move around an environment (Fig. 2.10) [24]. Given the static nature of The Swarm Garden, we take advantage of the absence of agent motion to centralize localization, however, this means that localization relies solely on the ability of the localization camera to perceive the modules' AprilTags, constraining the size of the swarm to the field-of-view angle of the camera. This serves as a centralizing element that can possibly act as a single point of failure (i.e a module is cut off from the view of the localization camera and does not receive neighbor assignment), unlike Kilobots, where localization is decentralized.

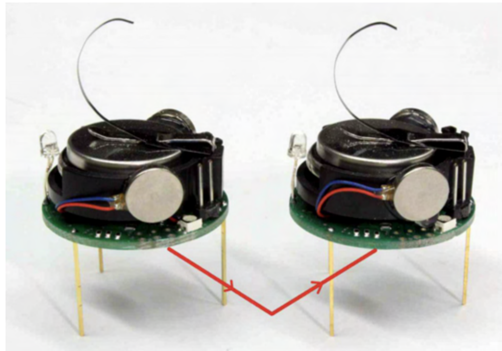


Figure 2.10: Infrared reflection path for inter-robot communication in Kilobots [24].

Figure 2.11: Example OpenMV output from localization camera (right). The Euclidean distance between AprilTags is calculated and threshold values are used to assign adjacent and non-adjacent relative positions.

Another consideration for localization is the configuration of the swarm in terms of adjacent module placement. Originally, The Swarm Garden was intended to be built in 3x3, square grids where there are straight rows and columns. However, as discussed in Section 2.2.3, a staggered grid configuration was used for the exhibit, causing the maximum number of adjacent neighbors to drop to 6 modules and the absence of true `'right'` and `'left'` neighbors to appear in the swarm (Fig. 2.12). We account for these directional changes in the development of interaction modalities that propagate to both the `'top[left/right]'` and `'bottom[left/right]'` neighbor in the event we want the system to express a visual response to a true left or right motional input from the operator (i.e pointing an arm straight to the right).

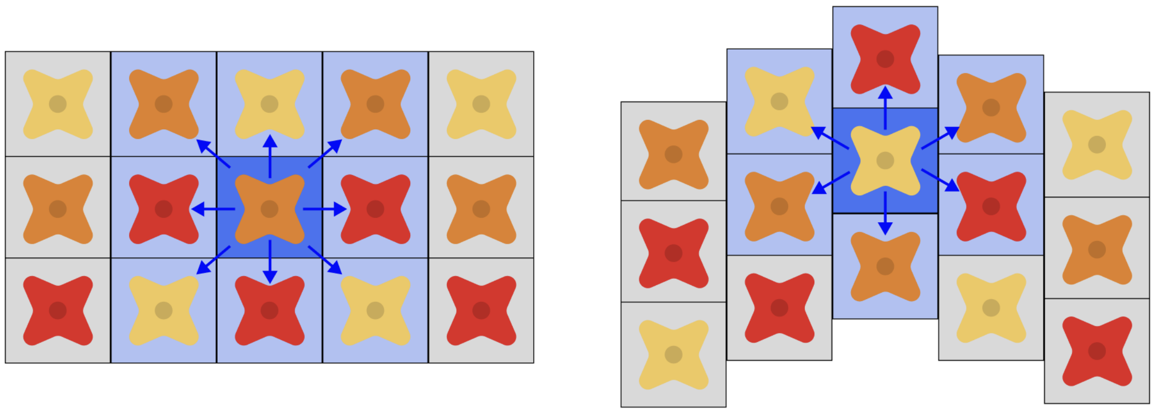


Figure 2.12: Adjacent neighbor assignment for square grid configuration (left) versus staggered grid configuration (right).

2.2.4 Message Content & Propagation

Communication within the swarm is facilitated through a standardized messaging system that utilizes string-based message parsing for coordinating state variable updates among individual modules. Each message is deliberately structured, containing essential information parsed by the receiving module. The message format for the propagated messages is defined as follows:

```
"[messageType] [prev_senders] [msg_id] [variable:content][...]"
```

Example 1 (all fields): `"LEDColorUpdate 4,1,7 124187 color:3"`

Example 2 (dummy `prev_senders` and `msg_id`): `"neighborsUpdate X X top:1"`

Messages are divided into four parts: `messageType` (Table 2.1), `prev_senders`, `msg_id`, and `content`. The content is further segmented into key-value pairs, such as "color:1" or "direction:right", and is maintained in a dictionary format. This structured approach ensures that modules can easily parse incoming messages, extracting relevant information for updating module state variables.

Table 2.1: List of message types and their usages for propagating state changes.

Propagated Messages Command Table	
messageType	**Usage**
`neighborsUpdate`	Updates `neighbors_list` based on the current placement of modules (only sent from the localization camera behind the swarm)
`modeUpdate`	Updates current mode / interaction modality (only sent from Central PC to all modules simultaneously)
`LEDColorUpdate`	Updates on-board LED color and propagates to all neighboring modules in `neighbors_list`
`LEDColorDirectionUpdate`	Updates on-board LED Color and propagates to neighbor modules in a specific direction (sends to modules in `neighbors_list` with a specific relative position)
`stripUpdate`	Updates LED NeoPixel strip color and propagates to all neighboring modules in `neighbors_list`
`stripDirectionUpdate`	Updates LED NeoPixel strip color and propagates to neighbor modules in a specific direction (sends to modules in `neighbors_list` with a specific relative position)
`bloomUpdate`	Triggers bloom mechanism (`bloom` $\rightarrow$ *retract* or `unbloom` $\rightarrow$ *flatten*) and propagates to all neighboring modules in `neighbors_list`

We define a simple *Listen-Handle-Forward* propagation scheme (Fig. 2.13) to allow modules to resolve their state based on incoming messages and decide whether or not to forward messages to neighbors. Different modes activate different listeners for corresponding `messageTypes`, allowing us to achieve particular interaction outcomes and visual feedback per each mode. We embed listeners for "self-commands" into the modules as well to handle state variable updates to singular modules for debugging and testing purposes as well as for modes like *Auto-Showcase* (Section 2.3.3), where we leave the modules running a non-interactive display to purely observe the hardware mechanism / buckling sheet technology. By employing this standardized messaging system, our swarm achieves efficient and targeted communication, enhancing overall system coherence and adaptability.

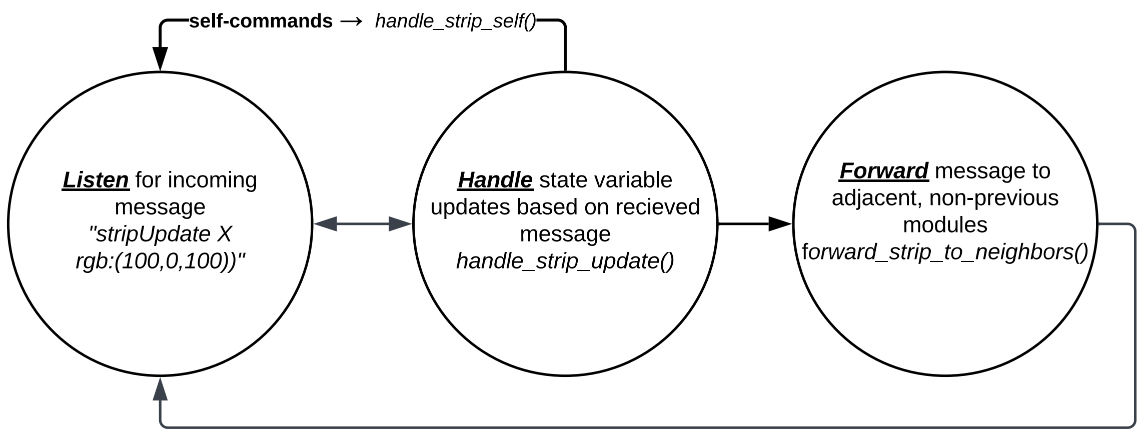


Figure 2.13: *Listen-Handle-Forward* propagation scheme for example message `'stripUpdate x rgb:(100,0,100)'`. Some handlers require listening to remain on to stay responsive during longer state updates (i.e slower blooming mechanism can be interrupted by other commands). Self-commands are not forwarded and return to listening after making the singular state update.

2.2.5 Message Redundancy & Swarm-Level State Resolution

To prevent infinite message loops between adjacent neighbors, all previous senders of a particular message are sent along with that message, denoted as `prev_senders`. As a message propagates along the swarm, modules will update their state accordingly with the corresponding handling function, then append their `module_ID` to the `prev_senders` of the current message before forwarding it to neighbors who are not previous senders of that message. We also check if the incoming message already has the `module_ID` of the receiving module and break the propagation if this is true. In this way, we prevent backwards propagation and infinite message loops as well as enable total state resolution of the swarm (Fig. 2.14). Messages that are uniquely sent by the Central PC or the localization camera (i.e `neighborsUpdate`) do not require this sender history, as the modules do not contain these devices as neighbors in their `neighbors_list`.

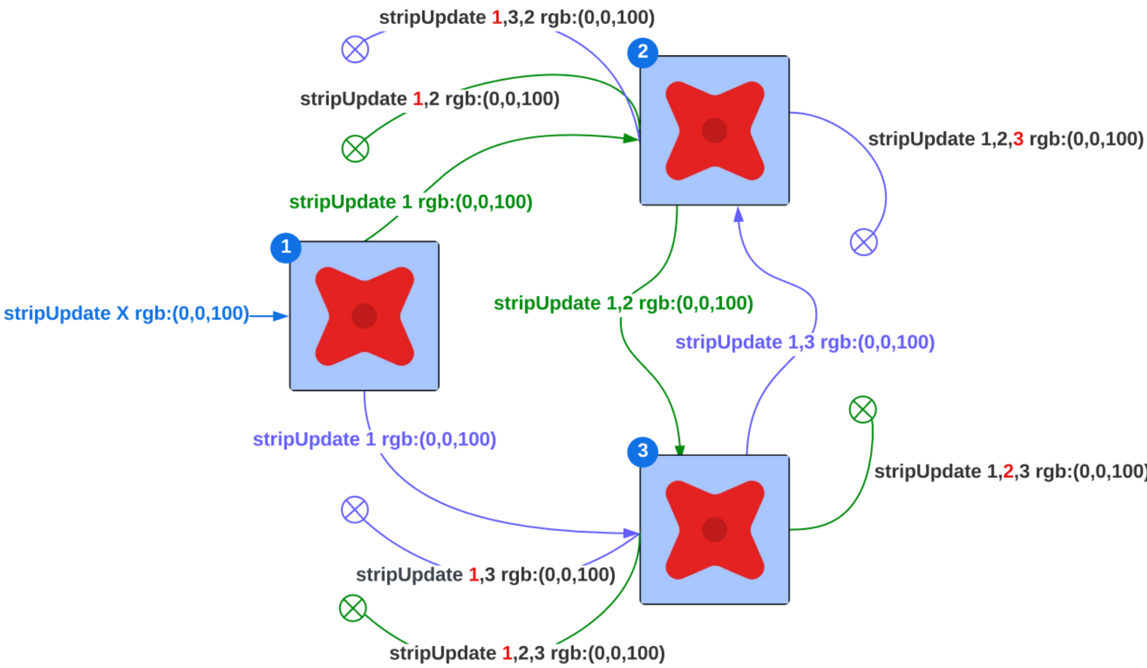


Figure 2.14: Message propagation displayed among 3 adjacent neighbors. The starting message, `'stripUpdate X rgb:(0,0,100)'`, is sent from the Central PC to Module 1. Connected arrows denote a successfully forwarded message, open-ended arrows denote a message break (end of propagation). Purple and green denote two different pathways from the initial message sent to Module 1. Module 2 and 3 pass the message between each other once as a redundancy check. This finally breaks the propagation pathway now that all modules have become previous senders of the message.

To further optimize message looping prevention, we also utilize a `msg_id`, which is a randomly generated 6-character number string for every unique message sent throughout the swarm. During every state update handler, each module checks whether the incoming message is the same as the last received message. This way we avoid consecutively spammed messages between neighbors. For example, we can avoid the redundancy check in Fig. 2.14 by first checking if the incoming message is unique.

Overall, by employing the techniques of keeping a history of previous senders attached to messages with unique identifiers, we reduce overall latency in achieving swarm-level state resolution and convergence where we would otherwise see a rate of redundancy checks and message ping-ponging that would cause slower swarm-level state resolution. In Algorithm 3 and Algorithm 4, we provide an example handler function and its corresponding forwarding function to show how we employ these checks to successfully make state updates without unnecessary forwarding behaviors that would compromise the speed of swarm-level state convergence. The checks described in this section are highlighted in blue.

Algorithm 3: Handler Function: Handle LED Strip Update

```
Global: LEDStripColor, neighbors_list, curr_msg_id
Function handle_strip_update(data):
    message_type, sender_id_string, prev_senders, incoming_msg_id, content ←
      parse_message(data);
    incoming_rgb ← content["rgb"];
    if incoming_msg_id == curr_msg_id then
        return;
                                  ▷ Rejected, duplicate consecutive message
    foreach prev in prev_senders do
        if str(module_ID) == prev then
            return;
                                  ▷ Rejected, backpropagation to previous sender
    curr_msg_id ← incoming_msg_id;
    FADE_SPEED ← 0.005
    foreach color in fade_color(LEDStripColor, incoming_rgb) do
        for i in range(n) do
              ▷ Steps through color transitions until new LED state is achieved
            np[i] ← color;
        np.write();
        sleep(FADE_SPEED);
    LEDStripColor ← incoming_rgb;
    forward_strip_to_neighbors(neighbors_list, incoming_rgb, sender_id_string,
      prev_senders);
```

Algorithm 4: Forwarding Function: Forward Strip to Neighbors

```
Global: module_ID, curr_msg_id
Function forward_strip_to_neighbors(neighbors, incoming_rgb,
 sender_id_string, prev_senders):
    foreach neighbor in neighbors do
        if neighbor not in prev_senders and neighbor ≠ 'far' then
                    ▷ Check if the message has already passed through neighbor
            sendData ← "stripUpdate" + " " + sender_id_string + "," +
             str(module_ID) + " " + curr_msg_id + " " + "rgb:" +
             incoming_rgb;
            s.sendto(sendData.encode(),('255.255.255.255', 50000 +
             int(neighbor.id)));
```

2.2.6 Input Timing, Visual Feedback, & Cognitive Delay

Within our human-swarm system, we have varying visual feedback speeds for inputs into the system for each output state variable change. For example, we found that the speed to completely bloom, as mentioned in Section 2.1.3, was significantly slower than the rate of interaction (i.e a gesture/proximity trigger (interaction) from an operator is significantly faster than the stepper motor blooming procedure (output)). This delay in interaction input to visual output can lead to cognitive gaps as the operator becomes prone to poor swarm-level state estimation given the slow reaction speed, and may lose the context to utilize control methods intuitively [9]. To remediate this, we propagate blooming commands at the half-way point of either blooming mechanism (blooming or flattening) to create a visual cue that the input is translating into a deliberate propagation across the swarm. As a module is blooming, the subsequent module receives the command and starts to bloom only once the previous module is half-way through its state update, creating the effect that the state is being handed off to neighbors as a result of the interaction (Fig. 2.15). This effect becomes more obvious when applied to the larger swarm and remediates some of the cognitive delay associated with the slow blooming rate, however, to truly improve this, a faster blooming mechanism or more powerful motor would be needed to better match the rate of operator interaction / input. The logic for timing the forwarding of a bloom command is highlighted in blue in Algorithm 5.

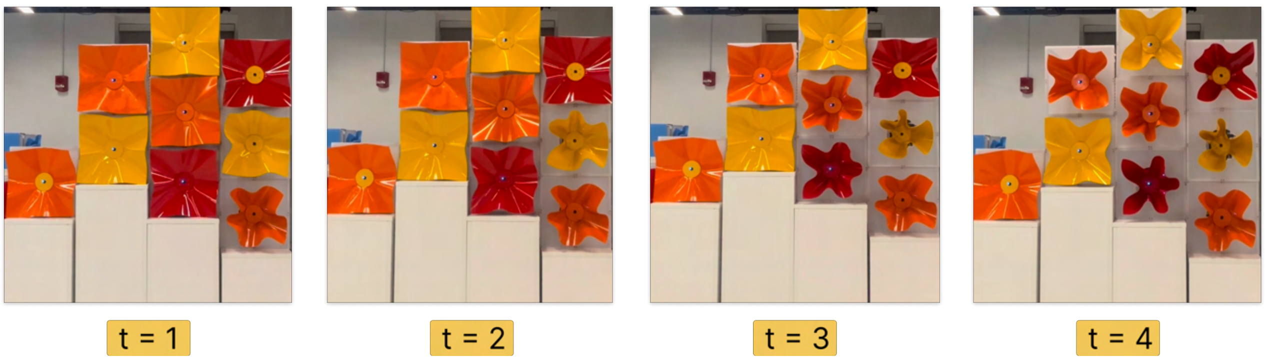


Figure 2.15: Example `bloomUpdate` command propagating from the bottom right module across 4 propagation steps. A visual gradient from bloomed to flat modules emerges to increase operator understanding.

There also exists a timing consideration for LED Neopixel strip command updates. In initial testing with propagating LED strip commands, the color of the receiving module would change instantaneously (apparent to the operators) and all consecutive propagated modules would also appear to change instantaneously. This contributed to a cognitive delay in that that we don't allow operators the opportunity to visualize the propagation pathway, and essentially jump to the end result of an interaction input. To ensure the propagation pathway of messages are recognizable to operators, we created fade-in-fade-out transitions between LED strip color updates to avoid abrupt, simultaneous multi-agent changes in state, adding a forced-delay on message propagation to emphasize the propagation pathways, thus reducing cognitive complexity and facilitating a stronger input to visual feedback connection (Fig. 2.16).

Algorithm 5: Handler Function: Handle Bloom Update

```
Function handle_bloom_update(data):
                                        ▷ Message parsing for bloom abstracted
                   ▷ Checks for redundancy and message loop prevention abstracted
    dist_from_stop ← tof2.read();
    if bloom is "unbloom" then
        while dist_from_stop < unbloom_thresh do
            listeningOn ← False;
            check ← upwards();
                        ▷ Returns False if interrupted by a different bloom command
            if check is False then
                listeningOn ← True;
                stop();
                return;
            end
            if ((unbloom_thresh + bloom_thresh) // 2) − 2 ≤ dist_from_stop ≤
            ((unbloom_thresh + bloom_thresh) // 2) + 2 then
                forward_bloom_to_neighbors(...);
            end

            dist_from_stop ← tof2.read();
        end
    end
    if bloom is "bloom" then
        while dist_from_stop > bloom_thresh do
            listeningOn ← False;
            check ← downwards();
                        ▷ Returns False if interrupted by a different bloom command
            if check is False then
                listeningOn ← True;
                stop();
                return;
            end
            if ((unbloom_thresh + bloom_thresh) // 2) − 2 ≤ dist_from_stop ≤
            ((unbloom_thresh + bloom_thresh) // 2) + 2 then
                forward_bloom_to_neighbors(...);
            end

            dist_from_stop ← tof2.read();
        end
    end
    listeningOn ← True;
    stop();
```

Figure 2.16: Example `stripUpdate` command propagating from the bottom right module across 4 propagation steps. The fading transition delay emphasizes the propagation pathway across neighbors to increase operator understanding.

2.3 Human-Swarm Interaction Modalities

In this section we will discuss the various interaction modalities The Swarm Garden is capable of. We mainly discuss proximity and vision based interaction modalities that are optimized for resource-constrained devices and briefly touch on the work of Yenet Tafesse, a collaborator working on a wearable device that is integrated into the human-swarm system. We will also cover mode switching and discuss the modalities still in experimentation (not stable enough for exhibition) and those utilized for public exhibition. We will touch on design inspirations for particular interactions as they relate to the nature-inspired / 'organic' quality we aim to embed within The Swarm Garden. Videos of each of the interaction modalities used in the public exhibition can be found in Appendix A.

2.3.1 Mode Switching

To switch between the interaction modalities, we utilize the Central PC to update the `mode` of all modules simultaneously. While we originally intended for mode switching to occur autonomously and hierarchically by assigning mode priorities and creating a subsumption architecture as done in related works [6], we ultimately opted for this simple, "remote controller" approach given our level of environmental control over the exhibition and desire to manually delineate transitions in interaction modes for testing and user studies. However, the trade-off in employing the Central PC for mode switching, is adding another centralizing element that creates risk for a single point of failure within the swarm rather than keeping interaction transitions completely decentralized.

We originally programmed each module to constantly check how much time has passed since the last received message. If the module didn't receive any inputs for X seconds, it would run a function to reset its state variables to base values. In this way we allowed the swarm to self-stabilize over time by making the results of interactions transient rather than persistent, resulting in the swarm 'canvas' to continuously be open to new interactions from new visitors as they pass through the exhibit. However, in testing we found that this autonomous self-reset interfered with the experience of interactions too greatly, thus we embed state variable resets on every mode transition to make returns to base conditions deliberate and controlled. Different interactions require different levels of state update persistence and transience, so opting for this controlled method was optimal for the success of our human-swarm interactions, especially given the environmental control we have over The Swarm Garden as an exhibition experience.

2.3.2 Interaction Modalities in Experimentation

Proximity Interaction Exploration Mode

As discussed in Section 2.1.2, the Arduino Nicla Vision has an on-board proximity sensor. We utilize this sensor to create an exploratory mode for operators to discover the different possible interaction outcomes and collaborate with other operators to affect the swarm-level state. On initialization to the *Proximity Interaction Exploration* mode, all modules are randomly assigned with various probabilities, an interaction outcome type for each of the observable state variable changes (blooming, unblooming (flattening), and LED strip color changes) (Fig. 2.17). Through experimentation, we found that having all 36 modules be interactive-types would overload the message propagation system, causing commands to stack as well as the visibility of long-range propagation to be lost. Thus, we also allow modules to be assigned an idle, non-interactive type to serve as purely connective agents to observe propagation of state updates from the more selective interactive-type modules.

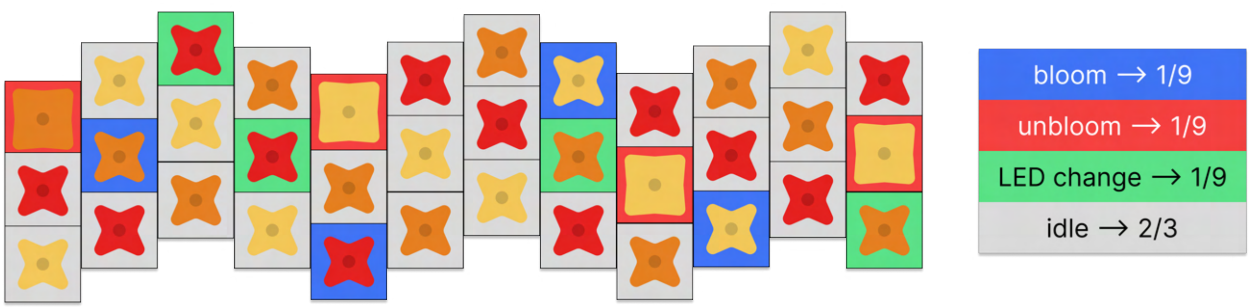


Figure 2.17: Example of assigned interaction types and corresponding probabilities.

When an operator enters the proximity range, bloom-type and unbloom-type modules bloom and unbloom respectively and propagate `bloomUpdate` commands to their corresponding type. LED-type modules propagate `stripUpdate` commands and pass along the color of their corresponding shimstock sheet element, allowing the colors of modules to blend and overlap as operators interact. Introducing stochasticity in this way allows operators the opportunity to engage with The Swarm Garden in an a-priori context that creates emergent outcomes for observation. This contributes to a sense of the 'natural' in that we aren't manufacturing or expecting a particular swarm-level outcome (Fig. 2.18). With each simulation of this mode, we achieve a unique configuration of bloom-type, unbloom-type, and LED-type modules that may or may not be conducive to the intuitiveness of the system or conducive to desired aesthetic and functional outcomes that operators are attempting to achieve.

Figure 2.18: Observed converged swarm-level state after a few minutes of interaction in *Proximity Interaction Exploration* mode. We see two main areas of color emerge (orange left, yellow right) and different neighborhoods of bloomed and unbloomed modules.

This mode remains in experimentation as there are still issues pertaining to timing and application to a swarm of this scale. First, to successfully allow messages to propagate through this mode, we enable listening for state variable updates at all times, even during active blooming and unblooming. Because of this, as LED updates are passed around the swarm, the blooming mechanism is interrupted during the fading color transitions, meaning consecutive LED updates can stall blooming for extended intervals of time. Also given the gap in reaction time between LED updates and bloom updates as described in Section 2.1.3, there is some ambiguity for operators as to what swarm-level outcomes correspond to their inputs. Thus, as the visual feedback per interaction overlaps and blends from sustained, consecutive interactions, the success of the swarm diminishes quickly over time.

Swarm Vision Visualizer

The Swarm Vision Visualizer is a web interface built to observe the live video camera feeds of the modules. Using a JavaScript wrapper for FFMPEG [25] [26], we stream camera data to the website and offload computation to the Central PC, allowing operators to see how the swarm is interpreting their movements in real time. Each camera stream is linked to their unique listening port to ensure proper routing and forwarding of commands back to the corresponding module the interaction occurred on. We also note the potential benefits of reducing cognitive load on operators by providing more salient visual feedback and context for how the swarm is interpreting live interactions [10], making this web interface interesting for further use in future work with The Swarm Garden.

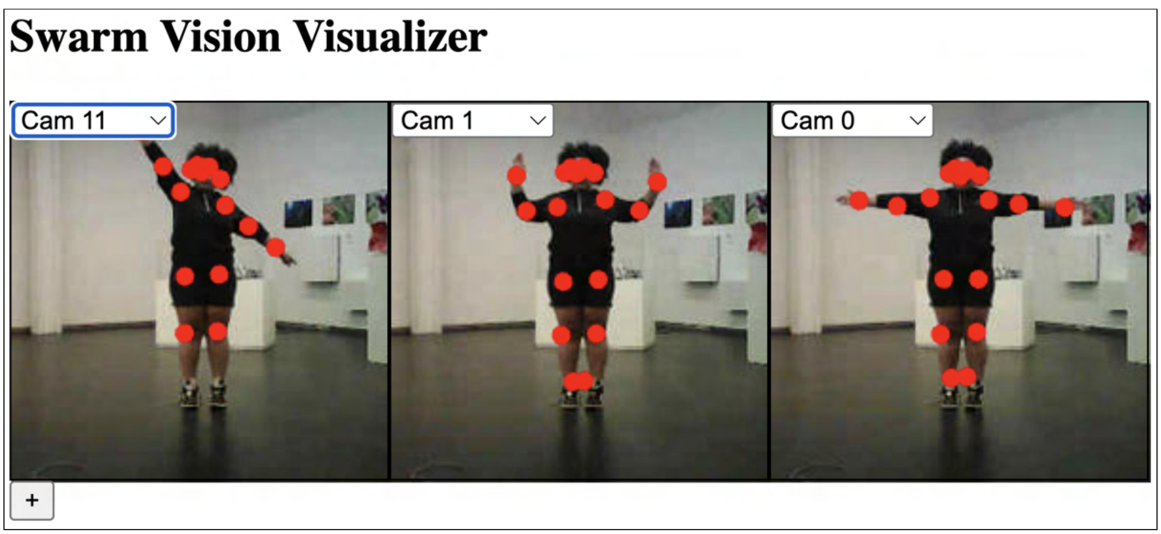


Figure 2.19: Swarm Vision Visualizer web interface where operators can add cameras to the view and observe live video streams running on each module with keypoints overlaid.

Pose Estimation & Gesture Recognition Mode

Pose estimation refers to the task of using a machine learning model to estimate the pose of a person from an image or a video by estimating the spatial locations of key body joints (keypoints) [27]. In our application of pose estimation, we use keypoints to define recognizable point gestures for various directional, and long-range interaction outcomes.

Figure 2.20: A failed pose estimation test via Edge Impulse using a multi-person scene. Despite having full body context, overlapping features are not accurately detected (left forearm and bottom of left leg). These inaccuracies and the size of the model encouraged a pivot away from running pose estimation on-device.

After experimenting with deploying a pose estimation model using Edge Impulse (ML development platform for edge devices / microcontrollers) there were issues with the size of the models generated by this software (Fig. 2.20). Also, given the resource constraints of the Arduino Nicla Vision, we found it unfeasible to store and run various models completely on-device. In addition, while Edge Impulse could generate keypoints on images passed in through their web interface (Fig. 2.20), their proprietary deployment process for this particular model made it impossible to run pose estimation live, directly on the Nicla. Thus, we offload the interactions for pose estimation to the Central PC via the Swarm Vision Visualizer (Fig. 2.19 and Fig. 2.22). Using the pre-trained TensorFlow.js model for pose estimation, PoseNet [27], keypoints for live human poses are generated. Through experimentation, we utilize the angles of the right and left forearm (wrist to elbow) to set ranges for recognizing pointing gestures for the 8 possible adjacent neighbor propagation directions (using relative positions from Section 2.2.3). The point gestures activate directional LED commands (`stripUpdateDirection`) with different colors to express left and right point gestures (Fig. 2.21).

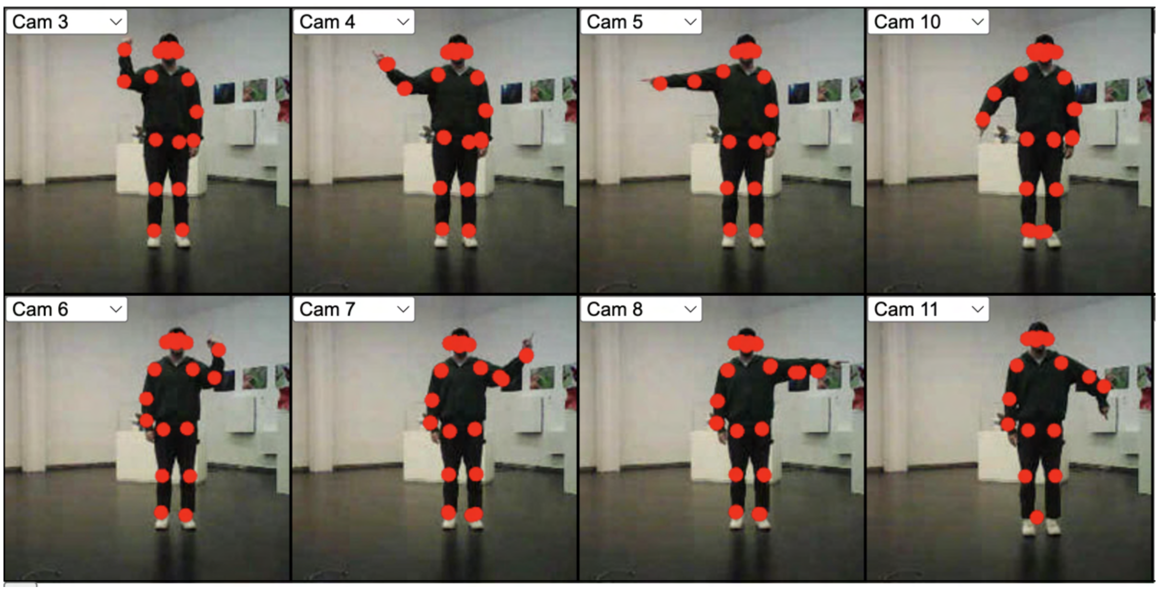


Figure 2.21: Top row: Right point gestures (relative to the operator) correlating to the following directional commands in order from Cam 3 to Cam 10 – `top`, `topleft`, `left`, `bottomleft`. Bottom row: Left point gestures (relative to the operator) correlating to the following directional commands in order from Cam 6 to Cam 11 – `top`, `topright`, `right`, `bottomright`.

This mode remains in experimentation as there were network issues with running multiple camera streams simultaneously on our Wi-Fi network, as well as finding consistently available IP ports with camera streams cutting out randomly and being unable to reconnect to the same IP, requiring a complete reprogram and reboot. In future iterations of The Swarm Garden this mode has the potential to utilize more salient / discrete gestures for more complex control patterns as well as applications in performance art and dance.

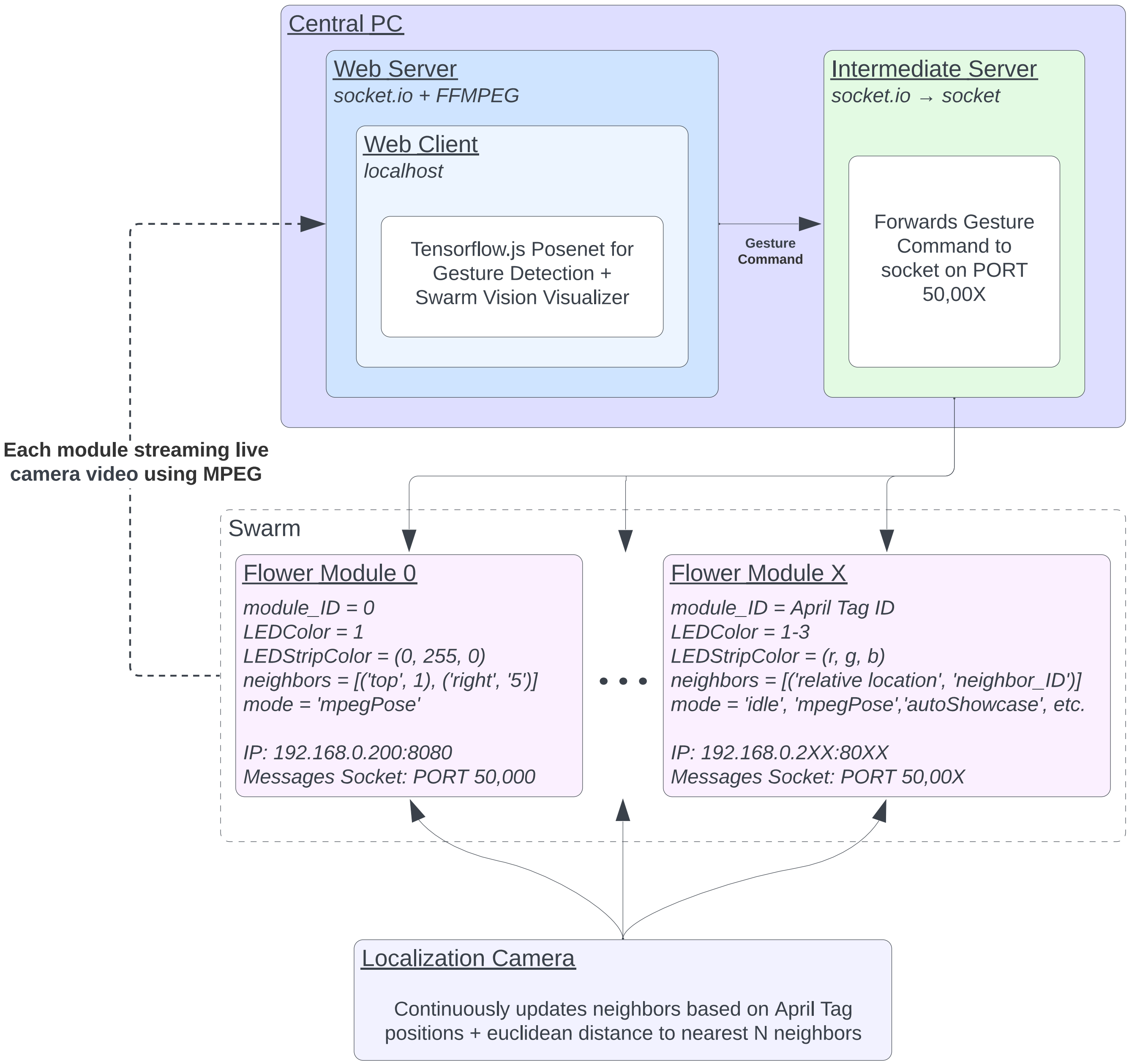


Figure 2.22: Architecture for sending live camera video feed to the Central PC, running the Swarm Vision Visualizer website for pose estimation and gesture recognition, and forwarding those gesture-based directional commands back to the swarm.

2.3.3 Interaction Modalities for Public Exhibition

Proximity Blooming Mode

The *Proximity Blooming* mode (Fig. 2.23) is a stable and simple mode used to purely display the buckling sheet blooming capability of the flower modules. On initialization to this mode, each module lights up with the LED color corresponding to its shimstock sheet color which remains constant for the duration of this mode. When an operator is close enough in proximity, the module either blooms or unblooms depending on checks to the time-of-flight sensor against the module's unique `bloomThresh` and `unbloomThresh`. When either threshold is hit, the motor simply changes direction to allow operators to continuously interact with the modules and observe the buckling sheet as it transforms between the two states (bloomed and flat). Operators also have the opportunity to observe the various patterns by which the flowers buckle based on each sheet's pliability and thickness.

Figure 2.23: *Proximity Blooming* mode. See videos 1.1 and 1.2 from the link to *The Swarm Garden Interactions* in Appendix A.

Transient Light Following Mode

The *Transient Light Following* mode (Fig. 2.24) is a stable and simple mode used to emulate a sense of embodiment in interactions with the swarm as we approach an ethos of the holistic and 'organic' in our interaction designs. When an operator is close enough in proximity, modules respond by lighting up with the LED color corresponding to their shimstock sheet color. However, when there is an absence of the operator within the designated proximity threshold, modules respond by turning off their LED strips. In this way as operators walk along the wall, a transient trail of light following their path illuminates, creating a shadow-like effect. The use of non-persistent state in this mode is an attempt to making The Swarm Garden feel like a living entity in that it has a perceived "attention" about who is nearby and subsequently acts on the recognition of nearby living organisms in the same way plants and animals do.

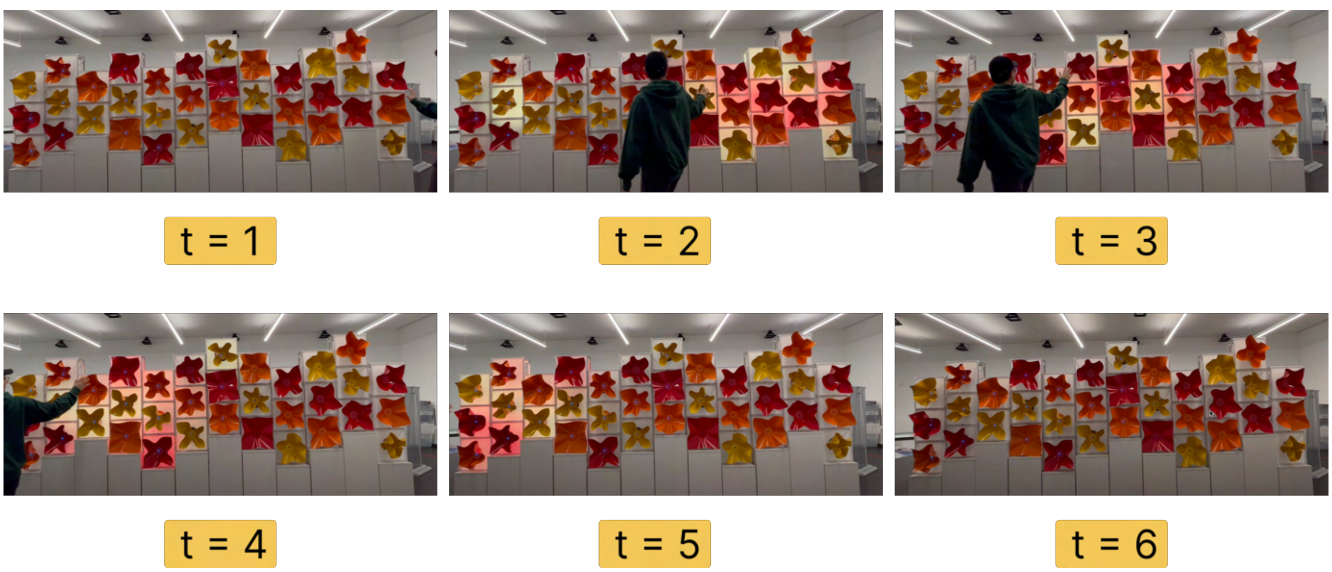


Figure 2.24: *Transient Light Following* mode. See videos 4.1 and 4.2 from the link to *The Swarm Garden Interactions* in Appendix A.

Auto-Showcase Cycle Mode

The *Auto-Showcase* mode (Fig. 2.25) is a non-interactive, pre-programmed cycling behavior of patterns where we can see the swarm perform various combinations of ordered and randomized blooming, flattening, and color changes. We essentially write animation snippets and have them traverse across the swarm from various start and end points, allowing exhibition attendees to get an understanding of the full range of aesthetic and functional capabilities The Swarm Garden has to offer. Used in this way, there is also the potential to use The Swarm Garden as a pedagogical tool, similarly to MIT CSAIL's use of the Distributed Robot Garden for youth computer science education by visualizing algorithms [20]. We envision the ability to simulate various swarming algorithms (i.e flocking, particle swarm optimization, etc.) as a near next step for The Swarm Garden, and *Auto-Showcase* serves as a first pass at running Python scripts of pre-set patterns and behaviors on the swarm without human input as an observational and educational tool.

Figure 2.25: From left to right, the *Auto-Showcase* animations in order: 1) Left-right zig-zag blooming with LED change. 2) Right-left zig-zag unblooming with LED change. 3) Row-by-row blooming with LED change. 4) Randomized LED change then randomized unblooming to reset cycle back to (1). See videos 3.1 and 3.2 from the link to *The Swarm Garden Interactions* in Appendix A.

Wearable Device Mode

The *Wearable Device* mode (Fig. 2.26) constrains all human-swarm interactions to come from proximity and gestural (via IMU) input to a wearable device with corresponding wearable-specific messages being sent across various wearable interaction modalities. The two main wearable sub-modalities are described in Table 2.2.

Through collaboration with Yenet Tafesse, the designer and owner of the wearable

Table 2.2: List of interaction modalities within the *Wearable Device* mode.

Wearable Device Interaction Modalities	
Mode	**Description**
Pulse Mode	Based on how long the operator covers the proximity sensor, the LED strips across all 36 will pulse at varying speeds (i.e short hold = slow pulsing, long hold = fast pulsing).
IMU Mode	Different orientations of the on-board IMU sensor of the wearable correspond to different LED strip commands with varying propagation styles (i.e arm straight up = choose a random module and propagate a pink LED color update command to all neighbors). This way as an operator engages in improvisational dance, various stochastic patterns emerge.

device's hardware and software design and implementation, we had the opportunity to observe an instance of integrating a new control method for interaction within the base swarm system proposed and built throughout this thesis. With handlers and forwarders for all state variables already built into the system, following the *Listen-Handle-Forward* scheme (Fig. 2.13), we had the existing infrastructure to easily add new listeners and handlers for the *Wearable Device* mode. This facilitated a straightforward development flow as we used the base-system's implementations to generate listener-handler pairs for each new wearable interaction and utilized existing state update forwarders. Through this this case of wearable device integration, we can validate the swarm system design and conventions I have created and adhered to across localization, neighbor management, and inter-module communication as generally intuitive for future work to be built upon the system beyond the current iteration of The Swarm Garden.

Figure 2.26: *Wearable Device IMU* mode: IMU's y-axis up example gesture (left), IMU's x-axis up example gesture (right). See video 2.1 from the link to *The Swarm Garden Interactions* in Appendix A.

Light Painting Mode

Building off of our integration of the wearable device into our human-swarm system, we introduce the *Light Painting* mode (Fig. 2.27 and Fig. 2.28) to explore a use case of The Swarm Garden as a painting-like, art-making tool. Upon transition to this mode, all modules bloom to allow visibility of all LED strip outputs across the swarm. The wearable device acts like a paintbrush with an array of pre-defined RGB values stored on the device acting as the paint palette. Using a close proximity threshold to the modules, when an operator's hand is close to the sensor, the module's LED strip turns on and off with the RGB value of the currently selected color from the color palette. This allows operators to paint and erase "pixels" as they work on their painting on the canvas. Through a counter-clockwise rotation of the right wrist where the wearable is attached, operators can cycle through the available colors.

On every rotation gesture, the swarm outputs the incoming color being selected from the palette for a few seconds before returning to the current canvas state (any previous progress on the canvas is saved and returns after this quick display for confirming the currently selected color). *Light Painting* serves doubly as a task-

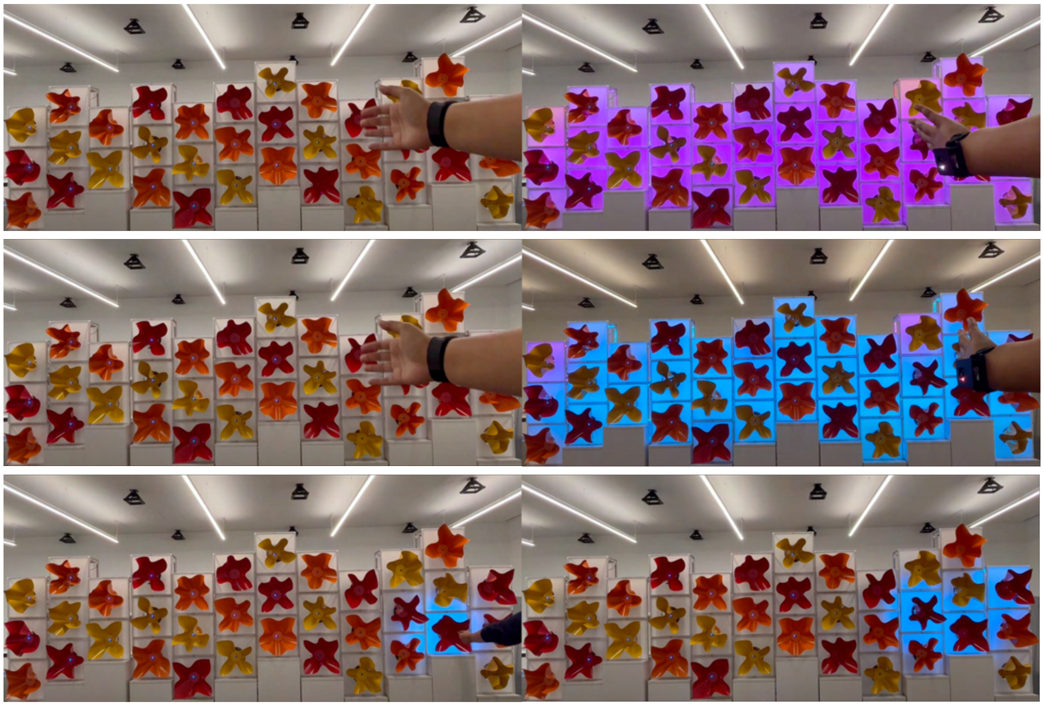


Figure 2.27: Example of consecutive command execution for *Light Painting* mode. From the top to bottom row: 1) Change color to purple. 2) Change color to blue. 3) Paint blue onto the canvas. See video 6.1 from the link to *The Swarm Garden Interactions* in Appendix A.

based, art-making modality as well as a good test of the cohesiveness of The Swarm Garden ecosystem in that both the swarm and wearable are actively perceiving and responding to input, requiring deliberate and well-defined control methods to create a seamless and intuitive user experience for operators.

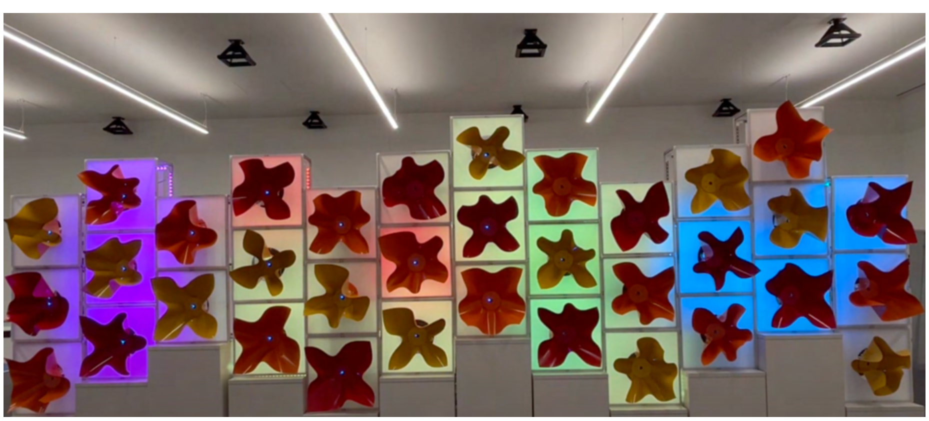


Figure 2.28: Final painting after using *Light Painting* mode.

Chapter 3

Evaluation and Results

3.1 The Swarm Garden Exhibition

The Swarm Garden was shown in a public exhibition on April 9, 2024 in the Lewis Arts Complex CoLab, where visitors were able to interact with system via the interaction modalities described in Section 2.3.3 (excluding *Light Painting*) (Fig. 3.1). After opening the exhibit with a run of *Auto-Showcase*, we cycled through 15-30 minute intervals of *Proximity Blooming* and *Transient Light Following.* A professional dancer and project team member, Azariah Jones, performed a live, human-swarm collaborative improvisational dance piece utilizing the wearable device midway through the exhibition, allowing visitors to observe the emergent behaviors of the The Swarm Garden in response to expressive, gestural input. Following the dance performance, visitors were able to try out the wearable device for a bit before returning to the non-wearable modalities. The exhibit was held for 3 hours with mixed media (posters, display cases, auxiliary television) displayed throughout the room to provide more background technical information, motivation, and context for the project.

Figure 3.1: Photos from *The Swarm Garden: An Interactive Architecture Exhibition* on April 9, 2024 (taken by Lori Nichols).

3.2 Audience Sentiment Analysis

We will now discuss our methods for data collection during the exhibit and analyze the audience response to The Swarm Garden. First, exhibit attendees were not all formally briefed on the interaction modalities to encourage independent exploration of the system, however many attendees reached out to team members for clarifications throughout. Exhibit attendees were able to optionally fill out a form for a word cloud (Fig. 3.2) asking participants: "Describe your experience with Swarm Garden in one word". Participants were also prompted to provide any other feedback on the exhibit in an open-ended text format. While we estimate a little over 100 attendees of The Swarm Garden over the course of the 3-hour exhibition event, 57 unique respondents submitted a word or multiple words (up to 3) to the word cloud and 21 respondents left additional comments on the form for general feedback and reflections.

The audience response was observably very positive, with many viewers engaging excitedly with the different modes and trying to figure out what the control inputs were, inspecting the hardware by touching the shimstock sheet or looking behind the swarm at the AprilTags and localization camera, and asking deeper questions about how the system operates, future applications, and project inspirations. While we didn't poll for attendee demographics, we saw a diversity of visitors from professors and students across disciplines (robotics, architecture, visual arts, dance, etc.) to non-affiliated adults, children, and passerby, all with varying degrees of technical expertise and backgrounds exploring and engaging with The Swarm Garden throughout the exhibition.

Figure 3.2: Word cloud of one-word responses from 57 respondents during The Swarm Garden Exhibition. Each participant was able to submit up to 3 unique words to the word cloud.

Looking at the word cloud in Fig. 3.2, we immediately note the general positive sentiment across the largest words, with the most popular words being "cool" (frequency = 9) and "interactive" (frequency = 8). We further segment the words in the cloud by their frequency in Fig. 3.3, displaying a histogram of words that have a frequency greater than 1 to get a sense of what words were mutually shared (at least between two respondents) among the respondents.

To more formally and objectively define the overall audience sentiment, we utilize Python Natural Language Toolkit's (NLTK) VADER (Valence Aware Dictionary for Sentiment Reasoning) sentiment analysis tools [28] to extract sentiment scores based on the words submitted to the word cloud. We particularly emphasize the scores of negative words in the context of The Swarm Garden such as "manmade", "crowded", and "dim" to provide more robust and context-aware sentiment categorizations. We found that the audience responded with overwhelmingly positive sentiment with 95.8% of responses being of positive sentiment and 4.2% being of negative

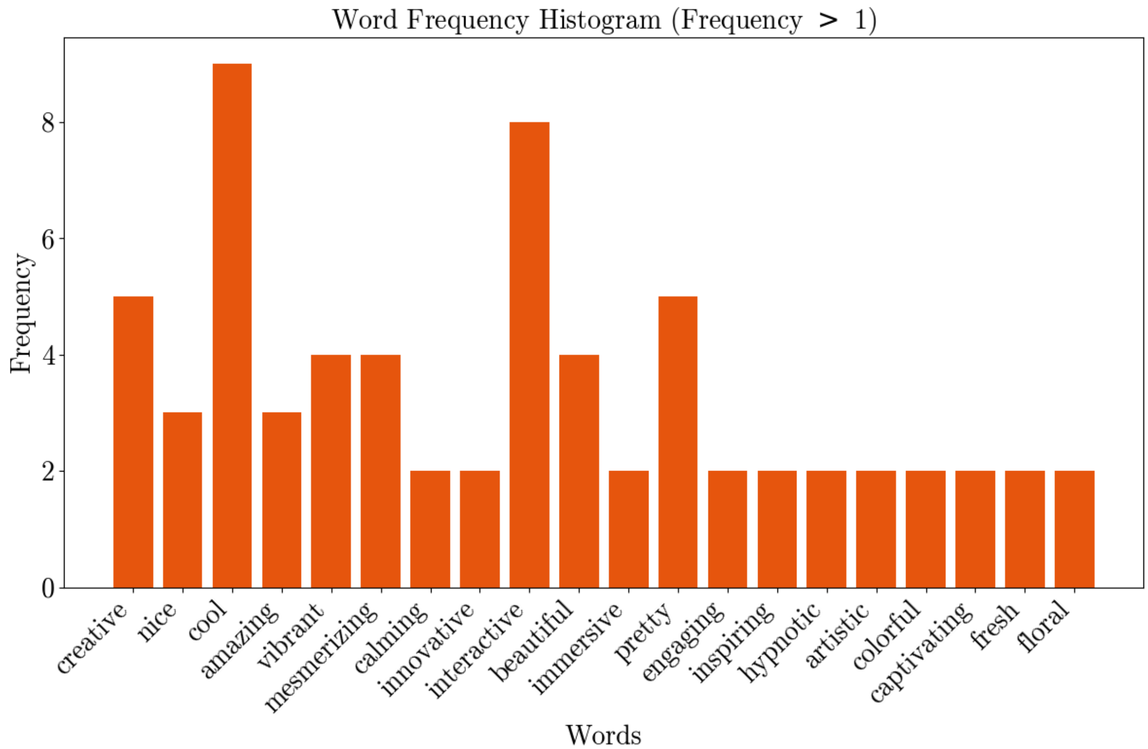


Figure 3.3: Frequency of repeated words (frequency > 1) from the word cloud.

sentiment (Fig. 3.4). We support this finding by the strong positive sentiments expressed by the majority of open-ended responses (19 of 21 open-ended responses were positive, 1 was a neutral question, and 1 was a constructive criticism / improvement suggestion), some of which including:

- *"Such a beautiful and vibrant experience!"*
- *"Such a unique display of robotics! The dance interaction is amazing!"*
- *"I liked how interactive the entire exhibition was. Great work!"*
- *"This is sooo cool, it was so amazing to watch and the technology is even crazier! great work and thank you for sharing"*
- *"I loved how interactive the project is! This is frankly amazing!"*
- *"Lovely visuals and great interactive experience"*

To extract more meaning from and analyze these widely positive responses, we introduce our own categorization for contextualizing the overwhelmingly positive sentiment in response to the exhibition. We have outlined 4 key themes as an attempt to understand what elements of The Swarm Garden particularly contributed to the largely positive sentiment response. We segment the one-word responses into the following emerging key themes we observed: *aesthetics*, *novelty*, *interactivity*, and *emotional impact*. Table 3.1 provides our definitions of each of these key themes within our categorization and examples of words that would fall into each type.

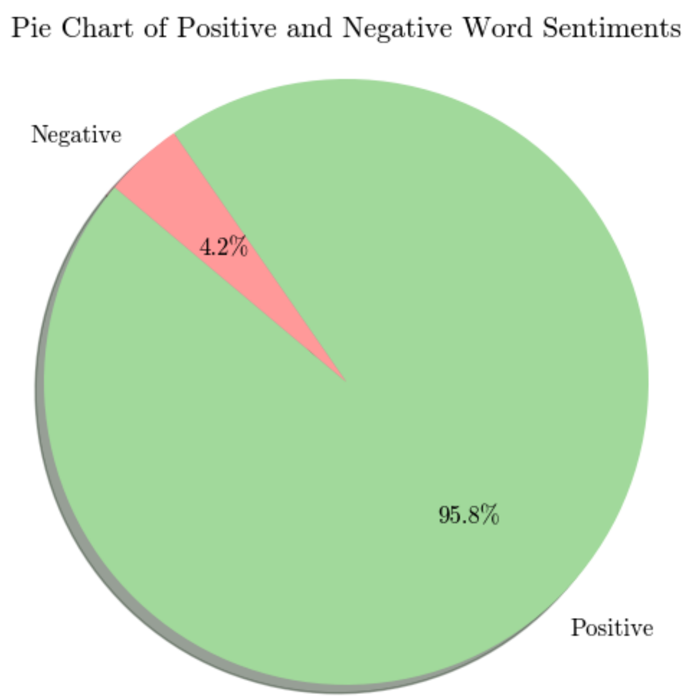


Figure 3.4: Pie chart of the sentiment across all one-word responses provided by respondents. Sentiment values were produced using NLTK VADER sentiment analysis tools `(SentimentIntensityAnalyzer)` [28].

After assigning each one-word response to a particular key theme, the distribution (Fig. 3.5) reveals that words relating to *aesthetics* were the most prevalent among positive sentiment responses, followed by *interactivity*.

Table 3.1: List of key themes of positive sentiment responses with definitions and example word associations.

Key Themes Categorization Table		
Key Theme	**Definition**	**Example Words**
Aesthetics	Relating to the look and feel of The Swarm Garden from a purely artistic / visual design perspective.	vibrant, floral, sleek, modernist, elegant, bioinspired
Novelty	Relating to The Swarm Garden's quality of being new, original, or unusual.	novel, innovative, unique, forward, groundbreaking
Interactivity	Relating to the interactive, responsive, or kinetic quality of The Swarm Garden experience.	lively, engaging, immersive, dynamic, stimulating, fun
Emotional Impact	Relating to the emotional or attitudinal response to The Swarm Garden.	joy, uplifting, peaceful, euphoric, inspiring, spiritual

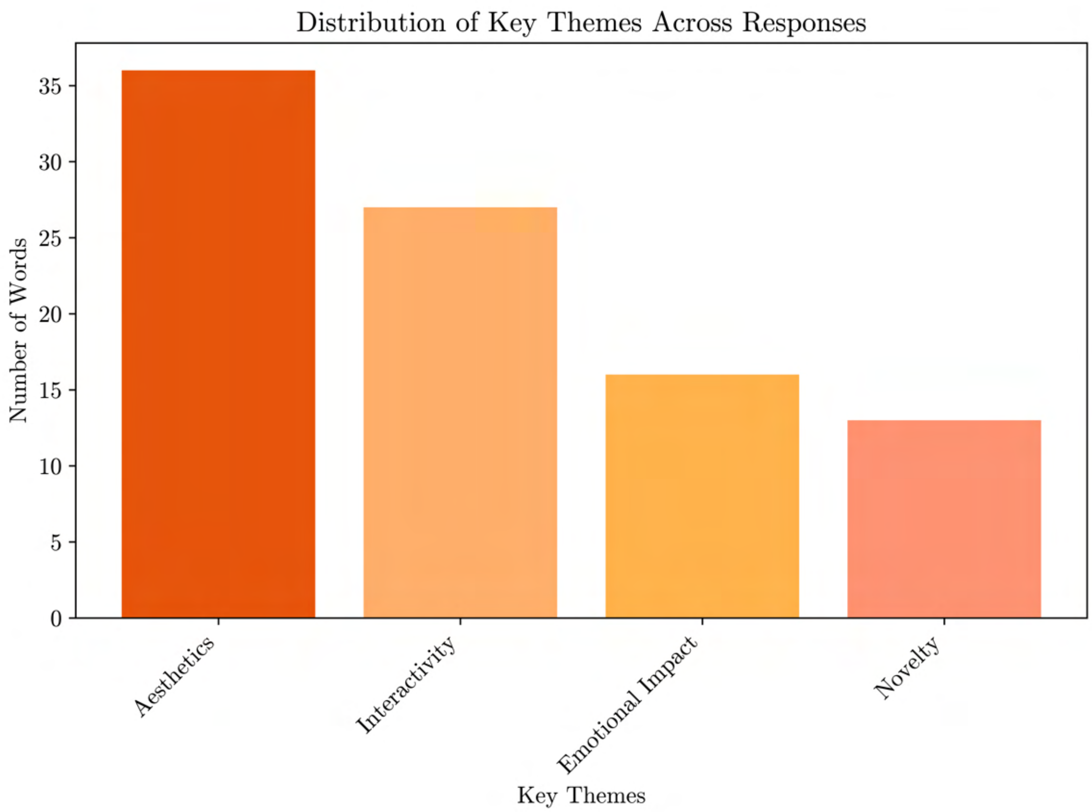


Figure 3.5: Distribution of one-word responses across the assigned 4 key themes.

We can now explore what the prevalence of *aesthetics* in the majority of positive responses possibly reveals qualitatively about The Swarm Garden. One possible line of reasoning is that the aesthetics of The Swarm Garden could have overshadowed the human-swarm interactions, not in a way that was visually overwhelming, but in the sense that people could have been drawn most to the look and feel of the flower modules and overall exhibition space rather than the human-swarm interactions. At the same time, it could be that the human-swarm interactions working in tandem with the aesthetics of the exhibition, particularly heightened the perception and essence of the aesthetics referenced by respondents (i.e bioinspired, floral, vibrant, etc.). We will need to more rigorously survey and analyze which particular modes were effective by both engagement and sentiment to validate these assumptions in the future, but we can critically evaluate The Swarm Garden relative to our own understanding of the system's shortcomings and the singular point of negative / constructive feedback from the open-ended responses to further contextualize and reason about audience sentiment.

Evaluating The Swarm Garden more critically, at its core, we could potentially attribute the lower references to *interactivity* to unsuccessful human-swarm interaction outcomes. We observed that a small sample of the exhibit attendees were confused about the control methods for interacting with the swarm, some of which trying various poses, motions, etc. and ultimately walking away from the swarm entirely before discovering the proximity sensor in the center of the modules. This cognitive gap between the expected output and attempted control (lack of control intuition) could have contributed to a sense of what one respondent described as a need to *"increase the harmony of the swarm"* and *"make it move more quickly"*. This coincides with the human input to visual feedback connection we attempt to address in our design, however, as mentioned in Section 2.2.6, we will need to update each module's hardware components for speed (i.e faster stepper motor) to truly close the input-output gap. Other possible solutions include embedding instructions on the swarm via more obvious iconic representations that users can recognize without having to recall them

[29] or utilizing the Swarm Vision Visualizer alongside the The Swarm Garden via an auxiliary display to provide more context [10] for operators to ramp up to control methods more intuitively and independently as well as remediate cognitive delays.

We also found an observable difference in audience engagement across the two cycling non-wearable modes, *Proximity Blooming* and *Transient Light Following.* At the moment of transition to *Transient Light Following* mode from *Proximity Blooming*, all of the LED's turn off in accordance with the behaviors described in Section 2.3.3. On every transition to this mode during the exhibit, we observed a sudden confusion in a some operators about what state variable of the modules was changing based on proximity, and without the more obvious blooming output (as the blooming mechanism has a physical, kinetic, and auditory presence than just the LED light), operators would walk away either out of disinterest or ambiguity until further instruction was given. We must also question whether the operators disengage with modes based on how inviting or approachable the interaction modality is. When all the LED's turn off on transition to *Transient Light Following* does the sudden darkness make The Swarm Garden less inviting? Is *Proximity Blooming* more approachable due to its more tactile / physical reaction to human input and consistent? Our results from the exhibition put these elements of The Swarm Garden into question, yet require further study with various groups of operators and different orderings of presented interaction modalities to reach conclusions on such research questions.

3.3 Human-Swarm Collaborative Dance Evaluation

To evaluate The Swarm Garden as a human-swarm interface for artistic applications we can examine the particular application exemplified by team member, Azariah Jones, in her use of The Swarm Garden for human-swarm collaborative dance performance and improvisation both on the day of the exhibit as well as during private testing and filming. As a professionally trained dancer of over 15 years across various styles (modern, Afro-modern, Carribean folk, ballet, hip hop) we look to Azariah's reflections of The Swarm Garden as a tool for movement-based art in the context of dance improvisation to evaluate the success of our human-swarm system in this particular context albeit isolated and ungeneralizable as we await further official user studies to make conclusive statements on how The Swarm Garden generalizes to dancers at large. Below we provide an excerpt from her reflections:

> "I think it made me very aware of the way I moved my individual body parts. Usually with dance, especially with contemporary specifically, I'm more focused on the pictures my entire body is making and capitalize on the flow and breath inherent to the style to move through movements rather than make individual pictures. I think using this made me hyper-aware of where my hand with the wearable was in relationship to the rest of my body. It definitely was isolating my wrist specifically, and when I performed I felt my brain categorize my right wrist and the rest of my body as two separate things. I wouldn't say it hindered my performance, as much as it revealed to me how unconscious I was about that specific body part in the larger frame of my movement. I think also because both sides of the experiment, so really being conscious of the reactions vs. ignoring them and just dancing, there was no negative, or loss I felt in my performance, just a hyper-awareness and reminder that my body has multiple working parts that are both isolated and connected at the same time that I think I will carry with me. Thinking about it this way, my elbow is like my wrist's "neighbor" and there were times when I let the movement propagate from the wrist and that was controlling everything and into the rest of my body..."
>
> - *Rate your experience using the wearable device on a scale of 1 (very ambiguous) to 5 (very intuitive): 5/5*
> - *Rate how well your input (gestures / dancing) correlated to your expected outputs from The Swarm Garden on a scale of 1 (matched very poorly) to 5 (matched very well): 4/5*

From this particular instance of improvisational dance with The Swarm Garden, we can gather that our human-swarm system with a wearable device control input was intuitive and that dance-based gestural movements correlated well with the expected outputs. Observing a dancer come to realizations around increased body-awareness is particularly interesting as a framing for The Swarm Garden not only as an expressive swarm, but as an embodied swarm. This heightened awareness for specific body parts (particularly those that are serving as control input) raises questions surrounding the potential for embodied, architectural swarms in application to dance. Could The Swarm Garden serve as a pedagogical tool in areas of dance education? How might using multiple wearable devices on various body parts simultaneously influence body-awareness? How do influences of The Swarm Garden on body-awareness and movement shift across different dance styles? Are some dance styles more likely to have successful outcomes in human-swarm collaborative dance improvisation than others? Overall, this team member evaluation guides us towards the exploration of The Swarm Garden in relation to conceptualizations of the body, expressive movement, and embodied human-swarm interaction.

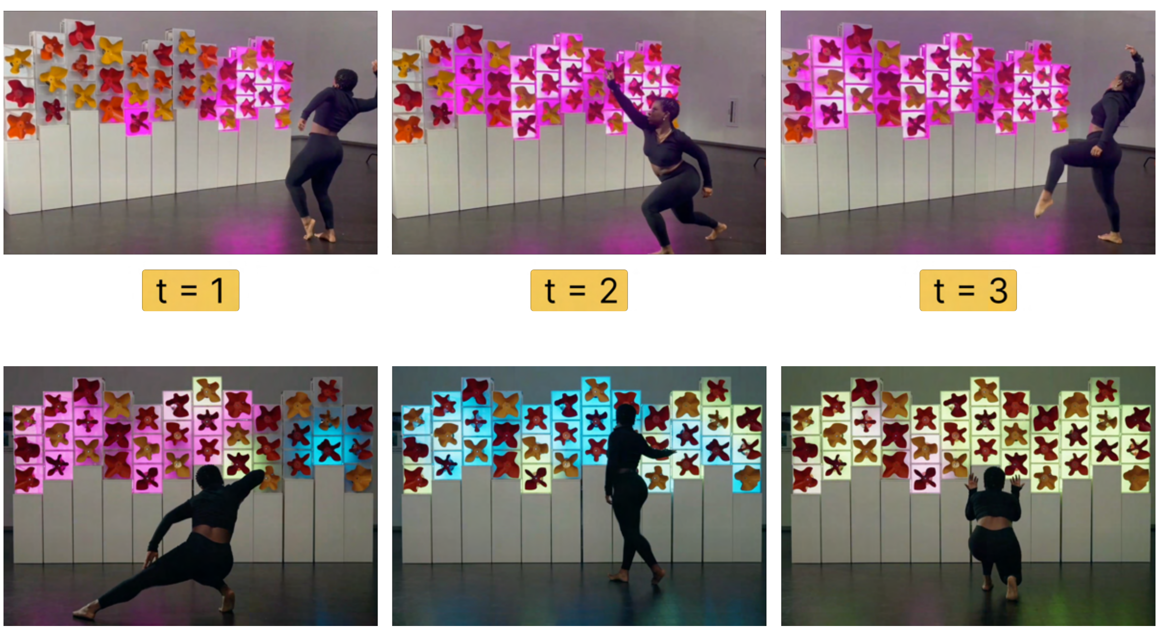


Figure 3.6: Human-swarm collaborative dance improvisation with The Swarm Garden. Top row: A consecutive chain of gestural inputs with corresponding visual outputs from the swarm. Bottom row: Various expressive movements / dance gestures and emergent outputs.

More rigorous study with various dancers of different styles and levels of experience will be required to answer questions about how The Swarm Garden would serve dancers and performance artists at large, however our Azariah's evaluation serves as a good barometer for the current development of The Swarm Garden as we work towards optimization of the human-swarm system in this particular context and application. Studies of this nature are under development within the Self-Organizing Swarms and Robotics Lab and will soon be carried out as we move towards future work on The Swarm Garden.

Chapter 4

Conclusions and Future Work

4.1 Conclusions

In this report, I have proposed and outlined the development of The Swarm Garden, a self-adaptive, nature-inspired architectural swarm display for human happiness and well-being. Bringing this work from concept to life as *The Swarm Garden: An Interactive Architecture Exhibition* has been an extremely rewarding and fulfilling process. Through the hardware fabrication of a full swarm of 36 modules to the software design of novel and intriguing interaction modalities to the overarching swarm-system and communication architecture, I've discovered new approaches and applications of human-swarm technology at the intersection of architecture, art-making, dance performance, and human expression.

The public exhibition has shown us the success of the current iteration of The Swarm Garden to carry out its purpose to making humans happy through beautiful, stochastic, and emergent human-swarm interaction through the overwhelmingly positive sentiment response. Despite hardware failures and constraints, the robustness of the swarm as a collective and overarching nature-inspired design brought out an innate aesthetic and functional cohesiveness that was reflected in the audience sentiment. In the same ways Anicka Yi's *In Love With The World* has posed questions about the role of robots in our society as agents of coexistence and collaboration, The Swarm Garden reveals the power of architectural swarms and expressive human-swarm interactions to further close the gap between human and machine in ways that are beautiful and holistic. We also observe the success of the human-swarm system in integrating auxiliary hardware (Wearable Device for Human-Swarm Collaborative Dance Performance and Improvisation), proving the system is capable of adopting more control methods and new modalities as they are built using the existing software infrastructure with the conventions and specifications described throughout this thesis. Likewise, we observe successes with the human-swarm collaborative dance performance and improvisation alongside The Swarm Garden, revealing the potential of The Swarm Garden to equally influence our perception of the body and expressive movement as much as we influence the swarm's "body" and expression through interactions.

In conclusion, The Swarm Garden succeeded in its objective of demonstrating an experimental nature-inspired interactive architecture exhibit comprised of 36 robotic flower modules that interact with people through self-organization and human-swarm interaction. By building with the intent to be scalable, responsive, and true to the components of human-swarm systems, we have achieved an iteration of The Swarm Garden that can serve as a test-bed for different interaction modalities for expressive swarms across different architectural, performance art, and general art-making applications. Though the results and conclusions of this thesis were very positive and exciting, there are many opportunities and necessities to optimize the existing iteration of The Swarm Garden especially on the fronts of fixing existing hardware issues, more rigorously testing outcomes of interactions across the discussed applications, and exploring system redesigns and improvements to scale and modify the system to suit various future, larger-scale applications.

4.2 Future Work

In terms of future work for The Swarm Garden, we must first address immediate next steps to carry out further studies and remediate the shortcomings of the current iteration. As mentioned in Section 3.3, we are currently in the process of getting approval for a study to test The Swarm Garden with the wearable for human-swarm collaborative dance performance and improvisation with a sample of dancers with different styles and levels of training. Besides this study we will also revisit our current module design to optimize for the speed of the motor, potentially reconsider the sensing / camera module used, and explore new blooming mechanisms to exploit the bistability of confinement. Furthermore, documentation will be written for the Self-Organizing Swarm and Robotics Lab on how to install, program, run, and test The Swarm Garden for future developers of the system.

Looking at future applications of The Swarm Garden, there are exciting considerations for how this expressive, architectural swarm can bring human happiness across many fields. First, as discussed in Section 1.3.2 and Section 2.3.3, there is a great opportunity for The Swarm Garden to serve as a pedagogical tool in the computer science education context for algorithm visualization as well as in the dance education context for building body-awareness. We also envision The Swarm Garden as a human-swarm system for performance art and dance by embedding our system into performance environments. We imagine this architectural swarm built into stages and performance venues that provide stochastic lighting and kinetic outcomes that are emergent, beautiful, and perhaps reach the ethos and intent of artistic works more intimately through this approach.

Finally, given the context of this project as a collaboration with the Form Finding Lab in the Civil and Environmental Engineering department, despite having many directions we can take The Swarm Garden, we will focus our current efforts towards moving forward in applying The Swarm Garden as a large-scale, self-adaptive facade for architectural applications.

Chapter 5

Research Attestation

Animal and Human Subjects Research Attestation

- In the case that no animals are used: Attestation that no animals are being used for this project.
- In the case that animals will be used: Approval and authorization from IACUC, Compliant with all of the University's IACUC policies, https://ria.princeton.edu/animal-care-and-use.

I attest that no animals were used for this project.

- In the case that no human subjects are used: Attestation that no human subjects are being used for this project.
- In the case that human subjects are used: Approval and authorization from the IRB, Compliant with all of the University's IRB policies, https://ria.princeton.edu/human-research.

Approval from the IRB has been attached on the next page.

Research Integrity & Assurance
Princeton University
Institutional Review Board
619 Alexander Road, Suite 102
Princeton, NJ 08540-6000

NOTICE OF APPROVAL

To:	Radhika Nagpal
From:	Institutional Review Board
Re:	IRB# 16722
Approved through (inclusive):	13-Mar-2027

14-Mar-2024

Dear Professor Nagpal,

On 14-Mar-2024, the IRB approved the following study.

IRB#:	16722
Title:	Swarm Garden: an interactive architecture exhibition
PI:	Radhika Nagpal

In conducting this study, you are required to follow the requirements in Princeton University **IRB Policy #207: Obligations of the Principal Investigator for Human Subjects Research**.

If you have IRB questions, please contact the IRB Office at (609) 258-0865 or irb@princeton.edu. If you have eRIA or other technical questions, please contact eria-irb@princeton.edu. In addition, our most frequently used resources are listed in the footer.

Thank you,

Daniel Notterman
Chair, Institutional Review Board

Chapter 6

Appendix

6.1 Appendix A: Code and Simulations

Codebase: https://github.com/Princeton-SSR/swarm-garden/tree/main

FFMPEG + Node.js: https://github.com/agsh/rtsp-ffmpeg/tree/master

Videos of Interactions: *The Swarm Garden Interactions*

All the code for this project is located in the first listed GitHub repository. The WEBSERVER and MIDSERVER are for running the Swarm Vision Visualizer on a central PC's localhost. The apriltags_neighbor_id.py file in NICLA is run on an OpenMV camera for localization. The module_interaction.py file in NICLA is programmed onto each module's Arduino Nicla Vision. NICLA/ADD_TO_DRIVE includes the libraries that must be added to each Nicla's storage to run module_interaction.py. All code pertaining to the wearable is found in WEARABLE. The REMOTE folder contains the autoshowcase.py file for running the pre-set animations for the *Auto-Showcase* mode. The open-source rtsp-ffmpeg for Node.js repository published under the MIT license is linked above.

6.2 Appendix B: Engineering and Industrial Standard

The independent project described in this thesis incorporated the following engineering and industrial standards:

- International units of physical quantities
  - International System of Units (SI)

SI units were used throughout the thesis.

- Programming languages and programming practices
  - Python, MicroPython, NLTK VADER sentiment analysis tools

Custom Python/MicroPython code was created for each interaction modality and for interfacing with the hardware components of each module. Python with NLTK VADER sentiment analysis tools was used to analyze user study results.

- Communication (hardware, protocols, and algorithms)
  - WiFi (IEEE 802.11)

A local Wi-Fi network connected to the Internet was used to facilitate communication within the swarm.

- Robotics/AI/ML based theses
  - Open source software: Edge Impulse, PoseNet / TensorFlow.js

Edge Impulse was used to create and train a test pose estimation model that ultimately was not used. TensorFlow.js / PoseNet was used for pose estimation in the Swarm Vision Visualizer.

Bibliography

[1] Air flower. [Online]. Available: http://www.liftarchitects.com/air-flower

[2] S. Adriaenssens, L. Rhode-Barbarigos, A. Kilian, O. Baverel, V. Charpentier, M. Horner, and D. Buzatu, "Dialectic form finding of passive and adaptive shading enclosures," vol. 7, no. 8, pp. 5201–5220, number: 8 Publisher: Multidisciplinary Digital Publishing Institute. [Online]. Available: https://www.mdpi.com/1996-1073/7/8/5201

[3] M. Alhafnawi, E. R. Hunt, S. Lemaignan, P. O'Dowd, and S. Hauert, "MOSAIX: a swarm of robot tiles for social human-swarm interaction," in *2022 International Conference on Robotics and Automation (ICRA)*, pp. 6882–6888. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9811723

[4] Leonard Lab Princeton, "Rhythm bots - prof. naomi ehrich leonard." [Online]. Available: https://www.youtube.com/watch?v=ScLB9rADdzo

[5] D. St-Onge, U. Côté-Allard, K. Glette, B. Gosselin, and G. Beltrame, "Engaging with robotic swarms: Commands from expressive motion," vol. 8, no. 2, pp. 1–26. [Online]. Available: https://dl.acm.org/doi/10.1145/3323213

[6] M. Alhafnawi, S. Hauert, and P. O'Dowd, "Robotic canvas: Interactive painting onto robot swarms." MIT Press, pp. 163–170. [Online]. Available:

https://dx.doi.org/10.1162/isal_a_00285

[7] A. Yi. Anicka yi: Intelligent floating machines inspired by nature | TED talk. [Online]. Available: https://www.ted.com/talks/anicka_yi_intelligent_floating_machines_inspired_by_nature

[8] Anicka yi | tate modern. [Online]. Available: https://www.tate.org.uk/whats-on/tate-modern/hyundai-commission-anicka-yi

[9] A. Kolling, P. Walker, N. Chakraborty, K. Sycara, and M. Lewis, "Human interaction with robot swarms: A survey," vol. 46, no. 1, pp. 9–26. [Online]. Available: http://ieeexplore.ieee.org/document/7299280/

[10] L. H. Kim, D. S. Drew, V. Domova, and S. Follmer, "User-defined swarm robot control," in *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. ACM, pp. 1–13. [Online]. Available: https://dl.acm.org/doi/10.1145/3313831.3376814

[11] M. Santos, G. Notomista, S. Mayya, and M. Egerstedt, "Interactive multi-robot painting through colored motion trails," vol. 7, p. 580415. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7806061/

[12] M. Alhafnawi, E. R. Hunt, S. Lemaignan, P. O'Dowd, and S. Hauert, "Deliberative democracy with robot swarms," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 7296–7303. [Online]. Available: https://ieeexplore.ieee.org/document/9981649/

[13] S. Ichihashi, S. Kuroki, M. Nishimura, K. Kasaura, T. Hiraki, K. Tanaka, and S. Yoshida, "Swarm body: Embodied swarm robots." [Online]. Available: http://arxiv.org/abs/2402.15830

[14] R. Simmons and H. Knight, "Keep on dancing: Effects of expressive motion mimicry," in *2017 26th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pp. 720–727, ISSN: 1944-9437. [Online]. Available: https://ieeexplore.ieee.org/document/8172382

[15] J. Alonso-Mora, S. Haegeli Lohaus, P. Leemann, R. Siegwart, and P. Beardsley, "Gesture based human - multi-robot swarm interaction and its application to an interactive display," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5948–5953, ISSN: 1050-4729. [Online]. Available: https://ieeexplore.ieee.org/document/7140033/footnotes#footnotes-id-fn3

[16] S. Ntzoufras, K. OungrInes, M. Liapi, and A. Papamanolis, "Robotic swarms in architectural design - a communication platform bridging design analysis and robotic construction," pp. 453–462. [Online]. Available: http://papers.cumincad.org/cgi-bin/works/paper/ecaade2020_334

[17] J. Gillin. HelioTrace robotic facade. [Online]. Available: https://www.dwell.com/article/heliotrace-robotic-facade-a7b58cd2

[18] A. Tabadkani, A. Roetzel, H. X. Li, and A. Tsangrassoulis, "Design approaches and typologies of adaptive facades: A review," vol. 121, p. 103450. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S092658052031030X

[19] Biofacades: Integrating biological systems with building enclosures. [Online]. Available: https://www.facadetectonics.org/articles/biofacades-integrating-biological-systems-with-building-enclosures

[20] Distributed robot garden | MIT CSAIL. [Online]. Available: https://www.csail.mit.edu/research/distributed-robot-garden

[21] Nicla vision | arduino documentation. [Online]. Available: https://docs.arduino.cc/hardware/nicla-vision/

[22] L. Stein-Montalvo, P. Costa, M. Pezzulla, and D. P. Holmes, "Buckling of geometrically confined shells," vol. 15, no. 6, pp. 1215–1222, publisher: The Royal Society of Chemistry. [Online]. Available: https://pubs.rsc.org/en/content/articlelanding/2019/sm/c8sm02035c

[23] AprilTag. [Online]. Available: https://april.eecs.umich.edu/software/apriltag

[24] M. Rubenstein, C. Ahler, N. Hoff, A. Cabrera, and R. Nagpal, "Kilobot: A low cost robot with scalable operations designed for collective behaviors," vol. 62, no. 7, pp. 966–975. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0921889013001474

[25] A. D.Laptev, "agsh/rtsp-ffmpeg," original-date: 2015-03-27T10:02:55Z. [Online]. Available: https://github.com/agsh/rtsp-ffmpeg

[26] FFmpeg. [Online]. Available: https://ffmpeg.org/

[27] Pose estimation | TensorFlow lite. [Online]. Available: https://www.tensorflow.org/lite/examples/pose_estimation/overview

[28] C. Hutto and E. Gilbert, "VADER: A parsimonious rule-based model for sentiment analysis of social media text," vol. 8, no. 1, pp. 216–225, number: 1. [Online]. Available: https://ojs.aaai.org/index.php/ICWSM/article/view/14550

[29] D. St-Onge, F. Levillain, E. Zibetti, and G. Beltrame, "Collective expression: how robotic swarms convey information with group motion," vol. 10, no. 1, pp. 418–435, publisher: De Gruyter Open Access Section: Paladyn. [Online]. Available: https://www.degruyter.com/document/doi/10.1515/pjbr-2019-0033/html?lang=en