

# Army Ant Inspired Adaptive Self-Assembly with Soft, Climbing Robots

A DISSERTATION PRESENTED

BY

MELINDA J. D. MALLEY

TO

THE SCHOOL OF ENGINEERING AND APPLIED SCIENCES

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

IN THE SUBJECT OF

MECHANICAL ENGINEERING

HARVARD UNIVERSITY

CAMBRIDGE, MASSACHUSETTS

JUNE 2020

©2020 – MELINDA J. D. MALLEY  
ALL RIGHTS RESERVED.



# Army Ant Inspired Adaptive Self-Assembly with Soft, Climbing Robots

## ABSTRACT

Social insects, including army ants (genus *Eciton*) self-assemble dynamic structures which allow them to perform more efficiently and achieve otherwise impossible feats. Army ant bridges, for example, grow and shrink according to traffic levels because they incorporate smart, flexible materials (i.e. living ants) and emerge from local behavior.

Similarly, self-assembled structures that respond to local stimuli could allow robots to adapt to new conditions when exploring unknown or rapidly changing environments. While many robot systems use rigid modules to create pre-determined structures, the work of my thesis, *Eciton robotica*, takes a new approach: inspired by army ants, soft robots create amorphous structures whose size and shape can adapt to the needs of the group.

*Eciton robotica* incorporates several novel features: the soft body, the gripping mechanism, communication through vibration, and the self-assembly algorithm. The robots' corkscrew gripper connects and disconnects easily to Velcro loop and supports loads in tension, shear and peel; it allows *E. robotica* robots to climb over other robots and attach to them at any point. In simulation, a simplified model of the robot was used to test a local control rule for adaptive self-assembly: robots stop and become part of a structure when stepped on. Using only local rules and information, simulated robots built and dissolved bridges in response to traffic and varying terrain. To implement this in hardware, the robots send vibration pulses as they walk, and sense this vibration if they are stepped on by other robots. Using vibration sensing and the local control rule, I used two *E. robotica* robots to form and dissolve adaptive structures in a 2D vertical plane.

Thesis advisor: Professor Radhika Nagpal

Melinda J. D. Malley

This work includes a number of firsts: The Flippy robot, an early robot module, is one of the first soft climbing robots, and highly capable though limited to Velcro and fabric based surfaces. The *Eciton robotica* platform, to my knowledge, is the first use of soft mobile robots for any collective behavior, including self-assembly, and the first case of *adaptive* self-assembly in a vertical plane.

# Contents

	TITLE PAGE . . . . .	<b>i</b>
	ABSTRACT . . . . .	<b>iii</b>
	TABLE OF CONTENTS . . . . .	<b>vii</b>
	ACKNOWLEDGEMENTS . . . . .	<b>ix</b>
<b>1</b>	<b>INTRODUCTION . . . . .</b>	<b>I</b>
	1.1 Motivation and Bio-Inspiration . . . . .	1
	1.2 Objectives . . . . .	5
	1.3 Contributions . . . . .	8
	1.4 Outline . . . . .	11
<b>2</b>	<b>RELATED WORK . . . . .</b>	<b>13</b>
	2.1 Self-Assembly in Nature . . . . .	13
	2.2 Self-Assembly in Robotics . . . . .	17
	2.3 Climbing Robots . . . . .	22
<b>3</b>	<b>ROBOT HARDWARE MODULE DESIGN: FLIPPY AND <i>ECITON ROBOTICA</i> . . . . .</b>	<b>27</b>
	3.1 Overview . . . . .	27
	3.2 Design Objectives . . . . .	29
	3.3 Locomotion . . . . .	30
	3.3.1 Body Design . . . . .	32
	3.3.2 Actuation . . . . .	39
	3.4 Gripper Design . . . . .	45
	3.4.1 Background . . . . .	46
	3.4.2 Gripper Design Requirements for Self-Assembly . . . . .	48

3.4.3	Design Concept Selection . . . . .	51
3.4.4	Testing of Gripper Designs . . . . .	55
3.4.5	Gripper Modeling . . . . .	60
3.4.6	Final Design and Validation . . . . .	62
3.5	Flippy: A Climbing Robot . . . . .	67
3.5.1	Fabrication . . . . .	67
3.5.2	Sensors and Electronics . . . . .	70
3.5.3	Control . . . . .	74
3.5.4	Climbing Experiments . . . . .	78
3.5.5	Climbing Over Other Robots . . . . .	84
3.6	Eciton Robotica . . . . .	87
3.6.1	Redesign and Adaptations . . . . .	88
3.6.2	Climbing over Other Robots - Final Results . . . . .	92
4	ADAPTIVE SELF-ASSEMBLY IN SIMULATION . . . . .	98
4.1	Overview . . . . .	98
4.2	Simulation Selection . . . . .	102
4.3	Simulated Robot Model . . . . .	104
4.4	Simulation Environment and Parameters . . . . .	107
4.5	Algorithm . . . . .	110
4.6	Experiments and Results . . . . .	111
4.7	Discussion and Conclusion . . . . .	120
5	<i>ECITON ROBOTICA:</i> SELF-ASSEMBLY BEHAVIOR IN HARDWARE . . . . .	123
5.1	Overview . . . . .	123
5.2	Detection of Robot Agents . . . . .	125
5.3	Self-Assembly Experiments . . . . .	130
5.3.1	Self-Assembly Algorithm . . . . .	131
5.3.2	Experiment Set-Up . . . . .	135
5.3.3	Results . . . . .	141
5.3.4	Discussion . . . . .	142
5.3.5	Conclusion and Future Improvements . . . . .	146
6	CONCLUSION . . . . .	148
6.1	Contributions . . . . .	148
6.2	Future Work . . . . .	150
6.2.1	Scaling Up to Swarms for Adaptive Self-Assembly . . . . .	150
6.2.2	Moving to 3D . . . . .	153
	APPENDIX A SIMULATION IN PYMUNK . . . . .	156

REFERENCES . . . . .	169
----------------------	-----

THIS THESIS IS DEDICATED TO MY FAMILY AND FRIENDS. I LOVE YOU ALL SO MUCH, AND THERE WOULD BE NO FLIPPIES WITHOUT YOU.

# Acknowledgements

First and foremost, I would like to thank my advisor, Radhika Nagpal, who provided invaluable feedback, patience, and support. She has been a friend as well as an advisor and an inspiration in many ways, fostering a lab environment where people care not only about research, but also about each other and making the world a better place.

Helen McCreery and Simon Garnier provided numerous insights on army ant behavior and feedback on the project. Working with them, especially with Helen after she joined the lab, and observing ant experiments in Panama heavily influenced my work and made me fall more in love with ants.

During the early stages of the project, Mike Rubenstein helped to advise the project; in addition to his help with the electronics design, many ideas and design concepts sprung from brainstorm sessions between myself, Radhika, and Mike. Bahar Haghighat's work on the final PCB design was essential, she is organized and perceptive, and I could always depend on her for helpful feedback. I have had the privilege of working with dedicated and talented students like David Becerra, Suproten Sarkar, Noon Farsab, and Lucie Houel. I am especially grateful to Lucie, who did excellent work in implementing and analyzing the final simulation experiments.

Beyond the army ant team, the Self-Organizing Systems Research group, past and present, has been an incredible source of ideas, feedback, comfort, silliness and friendship. I am grateful to Melvin Gauci for his help with the first IR touch sensor and for proving you are taller in the morning. Julia Ebert not only converted me to GitHub and VisualStudio but also gave my robot googly eyes. Jordan Kennedy was always ready to celebrate a success or take a walk, and, of course, gave us a lab mascot (Holly). Florian Berlinger, Nic Carey, and Daniel Calovi, Jeff Dusek, Justin Werfel, Nathan Melenbrink, Nils Napp, Kirstin Petersen, Lucian Cucu, and many others contributed in some way to this work and to my growth as a researcher. I am especially grateful to Julia, Melvin, Jordan, Florian, Nic, Daniel, Bahar, and Helen for filling the lab with serious research and serious shenanigans and a lot of baked goods.

I would like to thank my committee members Conor Walsh and Rob Wood for taking the time to evaluate my work and for their help and feedback over the many years on the project. The Flippy robot was inspired by soft robots made in Conor's class, and *E. robotica* has depended in part on Wood lab resources, particularly the laser-cutter and Instron; thus this work would not have been possible without them. Our lab administrator, Joanne Bourgeois has helped make sure purchases were as pain free as possible. Thanks to the Wyss Institute for supporting this research.

I am thankful for the Harvard robotics community, for all the people in the Wood, Howe, and Walsh groups who helped me to navigate SEAS and Harvard and who showed me how to be a graduate student and researcher. I especially owe Neel Doshi and Michelle Rosen for their support and friendship, and James Weaver for his help with 3D printing. Thanks to Jim

MacArthur for all his help with electronics, to Stan Coteau in the physics shop, and to Elaine, Maddie and Steve in the active learning labs for their assistance and resources.

My wonderful family, from my cousins and aunts and uncles - especially Faith, Grace, Dan, Margaret, Kathy, Kathy - to both of my grandmothers, my parents, and my sister, have always believed in and supported me. My parents taught me to love learning, and my sister has been my partner through many adventures. I am so lucky to have grown up with and be surrounded by truly good people.

Thanks to all of my friends and loved ones in Boston and beyond, who kept me (mostly) sane. Jesus Ruiz-Plancarte was with me and supported me before and throughout graduate school, and without him I would not have applied in the first place. Many thanks to old friends like Janelle Santiago and Anthony Jordan, who have stuck with me from afar.

Within Harvard, the GSAS Residence Halls gave me my first friends and a home for four years. I am grateful to so many: friends like Nathan Kwan and Eliza Gettel, my former residents, the amazing custodial staff, especially Miguel; our facilities manager, Bob; the B2 gang, especially Jackie Yun, Janet Daniels, and Ashley Skipworth who were always supportive of new ideas and positive change; my fellow RAs (and RRAAs), in particular Iosif Zhakevich, Somayeh Chitchian, Hayley Fenn, Ania Puszynska, Alex Zhang, Rodrick Kuate, Amy Tsang, and Tuo Liu, and many, many others. I learned so much about being a leader and building a community from being on this team.

To my incredible climbing partners and crew, particularly Christina Chang, Will Sprecher, Rebecca Levine, Elana Simon, Pallav Kosuri and Dominik Wild: thank you for all the beta and belays. I have especially depended on Dominik and Pallav for their friendship, guidance and support; they have both pushed me to be a better climber, a better graduate student, and a better person.

I am incredibly grateful to be part of the pole dance community in Boston, which has welcomed me and made me stronger in all senses of the word. Thanks especially to my teachers and to my Fly Together Fitness family - Abby, Alicia, Angela, Jenn, Jenny, Natalie, Quinn, Rachel, Tamar and Veronica continually support, inspire, and teach me.

Finally, thanks to Nik Lal and Boots, my pandemic quarantine and thesis-writing companions. Nik, thank you for sticking with me, for letting me steal your ideas, and for reminding me of the joy in research, machining, and pretty Solidworks models.

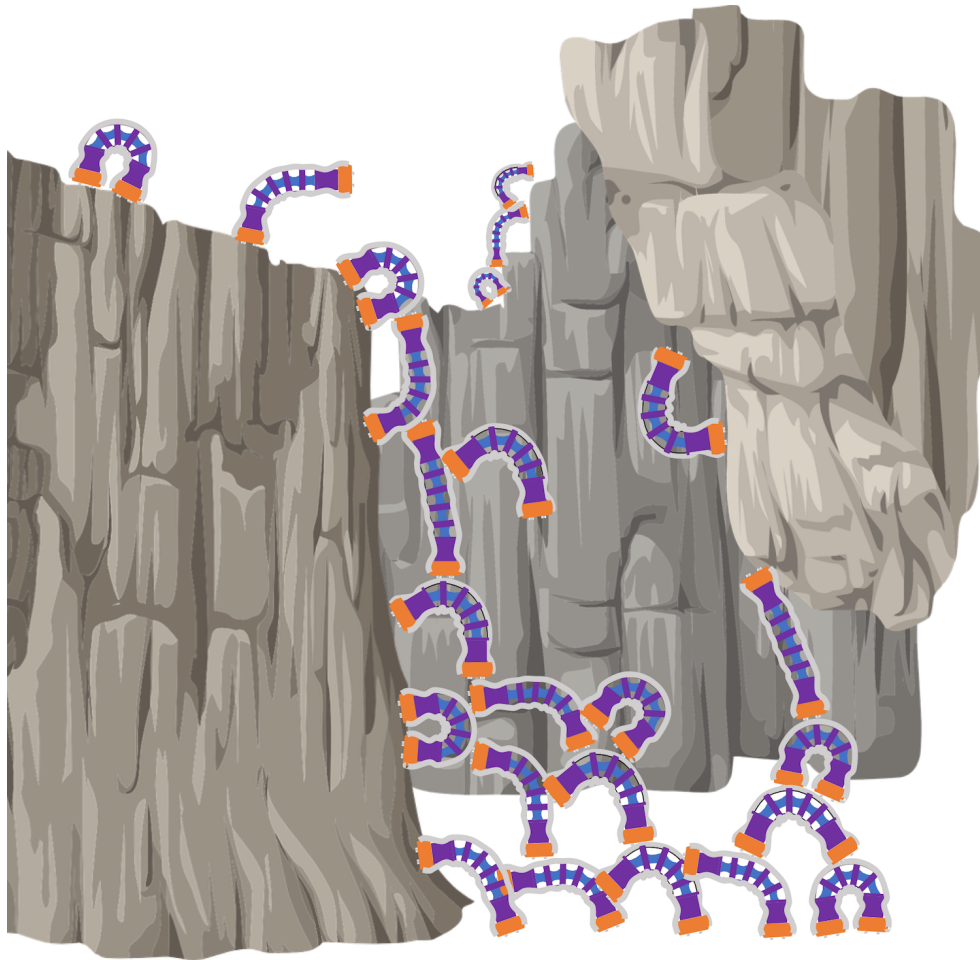


# Chapter 1

## Introduction

### 1.1 MOTIVATION AND BIO-INSPIRATION

Imagine a swarm of robots exploring a new territory or hunting for resources on another planet. While transporting cargo, they encounter a steep ledge, cliff or other rough terrain. Instead of getting stuck, they form a ramp, and traverse the obstacle. After the cargo has passed, the robots dissolve the structure and continue on their way. A storm comes, and they create a shelter to protect the cargo, without having to analyze and plan the structure, simply by reacting to local stimuli. They come across more resources which no robot can move by itself; a few robots form a chain and transport the resources back to the base.



**Figure 1.1:** A swarm of *Eciton robotica* could climb over difficult terrain in unknown environments.

While this may sound like science fiction, nature has already achieved this: social insects such as army ants, weaver ants, and fire ants, create living structures - bridges, rafts, pulling chains and more - from their bodies. These self-assembled structures enable insects to behave more efficiently and perform activities that are beyond the capacity of individuals. Army ants (genus *Eciton*) are an impressive example – ants form the building blocks of bridges, chains, and even their nests (called bivouacs), which functions as a mobile home, supporting their nomadic

lifestyle. While collecting prey, self-assembled bridges create short-cuts across rough terrain, allowing the ants to move quickly and efficiently. Unlike most structures built by humans with conventional construction methods, army ant bridges are adaptive and dynamic. Biologists have shown that these bridges not only conform to the geometry of the terrain, but also dynamically respond to the level of traffic<sup>1,2,3</sup>. At high traffic flows, larger bridges form; if traffic stops altogether, the ants in the bridge disassemble and return to other activities. This type of self-assembly allows insects to flexibly solve temporary problems and deal with emergency situations in their environment<sup>4</sup>.

Army ant and social insect colonies achieve self-assembly and other impressive feats through the collective action of individuals responding to local stimuli. This decentralized, swarm approach eliminates dependence on any one individual and allows the colony to quickly adapt and respond to new conditions. Similarly, small, simple robots working together in swarms may reduce cost and complexity of the system, eliminate single points of failure, and allow the system to adapt to new and changing environments<sup>5</sup>. By exploiting emergent behavior and different control rules, the same robot platform may be used to perform many different behaviors; the Kilobot robots, for example have been used for a large variety of purposes including to create defined shapes<sup>6,7</sup>, transport objects<sup>6</sup>, and form patterns mimicking morphogenesis<sup>8</sup>. Error tolerance may be found in numbers; if one or two robots fail in a large swarm, the group may still succeed<sup>5</sup>. This can save time for human operators and becomes especially important in dangerous environments where humans cannot recover the robots to conduct repairs.

Robot swarms perform an incredible variety of behaviors from aggregation and flocking to space exploration and object manipulation<sup>5</sup>. Many of these behaviors are inspired by social insects: the TERMES robots use bricks to build user-defined structures and adaptable ramps<sup>9</sup> in-

spired by termites. Kilobots and other robots transport objects collectively, a behavior inspired by ants<sup>10,11,5</sup>.

Robot swarms have also worked to tackle the challenge of self-assembly. Inspired by ants, Swarmbots, formed pulling chains and foraging paths<sup>12,13</sup>. Slimebot<sup>14</sup> takes inspiration from slime molds; the robot collective adapts and changes shape as it navigates through obstacles on a flat surface. While these two systems have created simple, but adaptive structures, the vast majority of self-assembling robots take their inspiration from cells, focusing on the creation of predetermined, lattice-based shapes<sup>15</sup>. Often using cube-shaped modules, robots reconfigure and self-assemble mostly into 2D shapes<sup>6,16</sup>, but occasionally also into 3D structures<sup>17</sup>. Some systems, inspired by the concept of programmable matter, consist of simpler, smart building blocks, and rely on their environment to assemble<sup>18,19,20,21</sup>. Others, such as the Kilobots<sup>6</sup>, M-Blocks<sup>17</sup> or SMORES<sup>22</sup> are fully autonomous, mobile robots. A few systems are self-healing - shapes can reform when disrupted<sup>6</sup>, but only Swarmbots and Slimebot change their structure in response to their environment.

While robot swarms and self-assembly systems are almost all composed of rigid robot modules, nature is full of examples of soft collectives, from cells to ants and other social insects. A few research groups have recognized the promise of soft robots: Inspired by cells, Cheney et al. evolved mobile robots composed of soft modular cubes in simulation<sup>23</sup> and Vergara et al. demonstrates the potential of soft pneumatic modules in hardware<sup>24</sup> to emulate morphogenesis. Emulating cells in an organism, these systems are centrally controlled, and modules function as building blocks, not fully autonomous mobile robots. Swarms of soft robots, which can conform to their environment and prove incredibly resilient<sup>25</sup>, thus are a natural progression but have yet to be realized.

In my thesis, I designed and developed the *Eciton robotica* system, combining the power of soft robots and collective behavior in a platform for adaptive self-assembly, inspired by army ants. Drawing from nature, I use soft, flexible robots equipped with a novel gripper and communication mechanism. By using simple, bio-inspired, localized control rules, robots in both simulation and hardware can create and dissolve structures which adapt to the environment and needs of the group. *Eciton robotica* pushes the boundaries of robotic self-assembly and soft swarms by demonstrating adaptive self-assembly in a vertical plane with fully autonomous, soft mobile robots.

## 1.2 OBJECTIVES

The overarching goal of this project was to develop a robot swarm capable of self-assembling amorphous, dynamic structures, such as the bridges and bivouacs formed by army ants. The research was divided into two areas: **hardware** i.e. the robot platform, and **algorithms** i.e. the local control rules for the robot system. Although described separately, these were developed concurrently.

Additional design considerations are outlined below:

- **Multi-agent systems:** Robot collectives can achieve tasks beyond the capacity of single robots and prevent single points of error by adding redundancy to the system. To create a swarm of robots, the robot modules must be scalable. I therefore prioritized price and ease of manufacturing.
- **Amorphous, Adaptive Structures:** While most robotic systems for self-assembly focus on predetermined, lattice-based shapes, ants and social insects create structures that



**Figure 1.2:** **Top** From left to right: A natural bridge formed by *Eciton hamatum*, an experimental set-up with the same species (image courtesy of Reid, Lutz, Garnier<sup>3</sup>), a hypothetical structure with the hardware platform *Eciton robotica*. **Center** Left: *Eciton robotica* in a ramp structure with two active robots. As robots climb over each other, they signal their presence using vibration. Right: Testing the novel robot gripper with robots in a chain structure. **Bottom:** Simulated robots form a structure that smooths traffic over rough terrain. Light colors indicate that the agents have gone into a “bridge” state and are stationary.

adapt to their environment. Structures are *function-driven* rather than *shape-driven*, and while not optimized for specific functions or environments, can adapt and serve a wider variety of uses. To allow for this variety and adaptability, I focused on attachment mechanisms and robot designs allowing flexible, amorphous structures.

- **Decentralized Control:** Centralized systems can allow for precise management of many robots, but create a single point of failure and generally require a controlled, known environment. I focus instead on decentralized control inspired by ants and other social insects which accomplish tasks with no central leader, and adapt to many different environments.
- **Limited Local Sensing and Simple Control:** Limiting the scope to local sensing and communication allows robots to quickly adapt and respond to new stimuli. Similarly, to prevent computation backlogs and allow for dynamic responses, I strive to keep the control of the robots as simple as possible. All robots are homogeneous, and assumed to be identical and follow the same protocols.
- **Bio-inspiration** Nature is a wealthy source of inspiration for robot behavior and interesting mechanisms. The central motivation behind this project is the self-assembly behaviors demonstrated by army ants and other social insects. I turn to nature for other design aspects of the robot hardware as well.

### 1.3 CONTRIBUTIONS

This thesis centers on my work developing the *Eciton robotica* project, and makes the three major contributions: **(1) a novel soft robotic hardware system for self-assembly, (2) a high level control algorithm for the construction of amorphous, adaptive structures, and the (3) implementation of this control algorithm in the soft robotic hardware system.**

#### 1. A novel hardware system for adaptive self-assembly with soft, flexible robots

For this project, I designed a flexible biped robot that can climb over Velcro and other fabric-based surfaces. The robot design includes a *novel corkscrew gripping mechanism*. The gripper is easy to control; once connected, it can stay connected without additional energy output; and it is strong in tension as well as peel, making it suitable for both robots with large moment arms, and self-assembled structures. By covering robots in Velcro, I created robots that could climb over each other and attach to each other at any point, enabling the creation of amorphous structures. Since flexible robots can be difficult to model, I used simple control mechanisms to exploit compliance where desired while creating predictable behavior.

My robotic hardware demonstrated several firsts. The Flippy climbing robot<sup>26</sup> was one of the first soft climbing robots, and, in my knowledge, still one of only two untethered and autonomous soft climbing robots. *Eciton robotica* is the first soft robotic for self-assembly. Of the few systems that have attempted adaptive self-assembly, ours is the first that we know of in the vertical plane<sup>12,14</sup>. Many of these firsts were achieved in part by the novel gripping mechanism combining corkscrew grippers and Velcro.



**2. A high level control algorithm for the construction of amorphous structures which adapt to the local terrain and traffic levels**

Previous work on self-assembly has mostly focused on the creation of pre-determined, lattice based shapes while rules for creating adaptive and amorphous structures have been left mostly unexplored. Through simulation, we demonstrated that robots following a simple control rule - “join when stepped on” - could create structures that varied based on the traffic levels and terrain, and could then dissolve those structures when they were no longer needed. To our knowledge, this is the first case of an adaptive self-assembly algorithm where structure size can adapt and respond to local stimuli (i.e. structures form, grow, shrink, or dissolve based on traffic levels). The high level control rules of this algorithm can be applied not only to our robot hardware and flipping bipeds, but also adapted to other robots with different gaits. In addition, the simulation provides further understanding into the behavior of the army ants; demonstrating the viability of this local rule for individual behavior.

**3. Implementation of the full algorithm in hardware**

As a final proof of concept, I built and programmed two robots which formed and dissolved a simple structure, following the control rules developed in simulation. To my knowledge, this is the first instance of adaptive amorphous assembly in a vertical plane and the first self-assembly system overall to use soft, flexible robots. The system can be expanded to larger, more complex structures with many robots. The system makes use of a novel communication mechanism through vibration where robots sense the presence of other robots by detecting vibration pulses. While vibration is used extensively in nature<sup>27</sup> and has been used in haptic feedback for humans, this is the first instance that we

know of using it for robot to robot communication.

The *Eciton* project vision was developed jointly with my advisor Prof. Radhika Nagpal, and Dr. Michael Rubenstein during his time as a research associate at Harvard. I have been the lead researcher in all aspects of this project, developing both the hardware and control algorithms; I conducted all of the mechanical design and fabrication of the robot, and performed all of the experimental work. For the electronics board design, I collaborated with Dr. Rubenstein and Dr. Bahar Haghighat. For the algorithmic work, I designed the conceptual framework and worked with two undergraduate students to develop the simulation and conduct initial experiments. The final simulation implementation was developed by Lucie Houel, an EPFL masters student who I supervised.

I informally collaborated with biologists Dr. Helen McCreery and Prof. Simon Garnier, who provided inspirational knowledge on army ant bridge behavior and helped conceptualize the algorithms for self-assembly. In March 2016, I travelled to Panama with the whole team (Nagpal, Garnier, McCreery, Rubenstein) to assist for a week with a new setup for experiments on army ant bridges. This trip greatly influenced my work.



**Figure 1.3:** Pictures from March 2016 trip to Panama: The author photographing an *Eciton hamatum* bivouac, underneath the bottom tree branch (Left). Dr. Helen McCreery and the author working to set up an army ant bridge experiment (Right).

## 1.4 OUTLINE

The thesis is divided into five parts:

**Chapter 2. Related Work** This section discusses self-assembly in nature, with a focus on adaptive self-assembly in insects, especially army ants, genus *Eciton*. I then give an overview of previous work in self-assembly in robotics, and discuss how this work builds on and distinguishes itself from prior work. Self-assembling robots share many design requirements with climbing robots. I therefore conclude the chapter with a review of climbing robots, including several which inspired the design of my robot hardware.

**Chapter 3. Robot Hardware Module Design** This section discusses the electro-mechanical design of the robot hardware module. Design requirements for the modules are presented which include climbing and attachment. The majority of the chapter is devoted to the development of the robot module - a soft, flexible biped that uses a flipping gait to climb. This includes a model for the locomotion and body design; design, modeling, and evaluation of the robot gripper; testing and evaluation of the module as a single climbing robot including the development of simple control rules; and finally, adaptations for the multi-robot system.

**Chapter 4. Adaptive Self-Assembly in Simulation** In this section, we create a simple model of the robot agents in simulation and use a physics simulator to explore self-assembly and dissolution of structures. We show that simple rules allow for the formation and dissolution of structures that adapt to traffic levels and varied terrains. The effects of different rules and parameters are discussed, as well as the future extensions to the work.

**Chapter 5. Implementation in Hardware** Here we demonstrate basic behaviors required for self-assembly including the vibration communication process for sensing fellow robots. We

then apply the local rules of the algorithm to form small, single robot structures. We discuss the current limitations of the systems and possible improvements.

**Chapter 6. Conclusion** This chapter concludes and suggests future directions, including new designs and considerations for creating structures in 3D - both in simulation and in hardware.

# Chapter 2

## Related Work

### 2.1 SELF-ASSEMBLY IN NATURE

Self-Assembly takes many forms in nature, from the crystallization of inorganic molecules to the formation of cells into organisms through morphogenesis<sup>29</sup>. In this chapter and for the thesis overall, I will focus on the self-assembly of structures made from living organisms, in particular structures made by social insects.

Many social insects self-assemble into bridges, rafts, nests and other structures with no centralized control, simply by following local rules<sup>4</sup>. These self-assembled structures enable the insects to behave more efficiently and perform activities that are impossible without the group.



**Figure 2.1: Self Assembly in Social Insects:** A) and D) Weaver Ant pulling and hanging chains, ©Chris Reid. B) Fire ants forming a raft<sup>28</sup>. C) Balling in Japanese Honey Bees, used to cook attacking hornets<sup>4</sup>. E) Small foraging bridge in army ants *Eciton hamatum* F) Larger bridge by *Eciton burchellii* ©Scott Powell. G) Close up view of an *Eciton hamatum* bivouac H) Zoomed out view of the same bivouac I) Larger exposed bivouac by *Eciton burchellii* ©Daniel Kronauer. E, G, and H were taken by the author in Panama.

Pulling chains, for example allow weaver ants to pull leaves together to create their arboreal nests. They also use them to transport heavy prey including birds orders of magnitudes larger than the ants themselves<sup>4,30,31</sup>. For army ants, building bridges and using their bodies to cover potholes in their path allows them to traverse obstacles and collect prey quickly<sup>4,1</sup>. Some insects use self-assembly to react to emergency situations: fire ants escape floods by forming living rafts<sup>28</sup>. Japanese honey bees even use self-assembly as a defense mechanism, forming “ovens” in which bees ball around an enemy hornet scout and cook it to death<sup>4</sup>. Examples of some of these structures are shown in Fig. 2.1.

Army ants, genus *Eciton* are especially good at these self-assembly behaviors; they form towers, bridges, and even their nests, called bivouacs, from their bodies<sup>4</sup>. Unlike most structures built by humans with traditional building methods, self-assembled structures naturally move, reshape and adapt to different needs and conditions. Army ant bivouacs, made of member bodies, are an adaptable, mobile home suitable to the ants’ nomadic lifestyle; army ants are constantly on the move as they follow their prey. Bivouacs can be made and remade in any environment - under a branch, or in hollowed logs - and can repair themselves if damaged by falling branches or other changes in the environment; even a researcher stepping into the nest only causes temporary destruction. Bivouacs can also perform thermoregulatory functions, maintaining an ideal environment for the queen and larva, while outside temperatures vary<sup>4</sup>.

While these bivouacs are impressive, army ants are perhaps even more famous for their bridges (e.g. Fig. 2.1e and f), which form along their trails and allow ants to collect prey and migrate more efficiently. Experiments have shown that army ants bridges not only conform to the geometry of the terrain, but also dynamically respond to the level of traffic<sup>1,2,3</sup>. At high traffic flows, larger bridges form; if traffic stops altogether, the ants in the bridge quickly disassemble

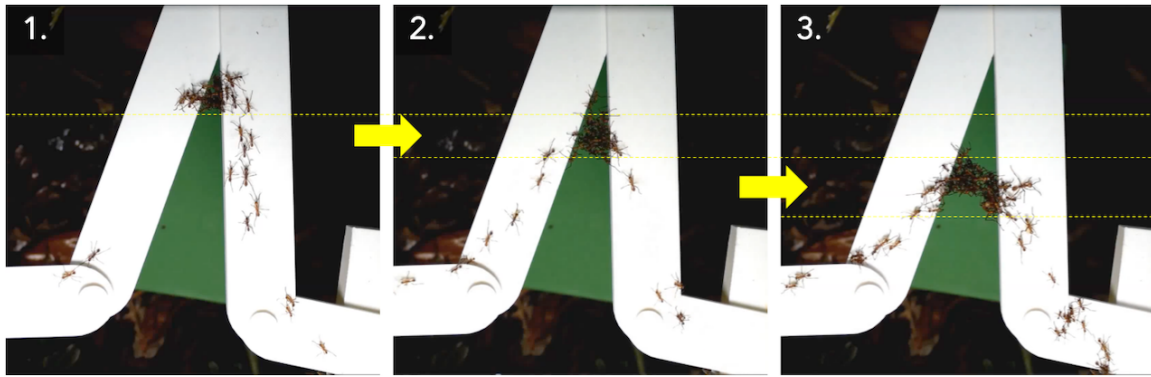


and return to other activities<sup>2</sup>. Adaptive self-assembly allows ants (and other insects) to flexibly solve temporary problems and deal with emergency situations in their environment<sup>4</sup>.

This kind of self-assembly could be useful in robotics applications, where robots similarly need to quickly adapt to unknown environments or emergencies. However, while we have witnessed ants and other insects build these structures and react to stimuli such as traffic levels, their methods, especially the local rules followed by individual ants, are not well understood. Garnier et al.<sup>2</sup> tracked the amount of time that individuals stayed in bridge structures as well as the local traffic flow of ants and showed that local traffic flow related exponentially with the time spent in bridge structures. The number of surrounding ants was also positively related to time spent in the bridge. While other factors may contribute to bridge building, we hypothesize that local traffic flow is a key local stimuli.

A central goal of this project was to test whether a simple response to sensed traffic flow would enable individual agents such as ants or, in our case, robots, to form stabilizing but adaptive bridges. We proposed to test one simple rule: Agents stay in place and become part of the “bridge” structure when they feel another agent step on them. The rule can be modified in various ways via different parameters - for example the relative time spent in the bridge per “stepped on” occurrence - some of which will be discussed in the simulation portion of this thesis in Chapter 4. However, the general principle seems to explain much of the ant behavior. In further experiments with *Eciton hamatum*, Reid et al.<sup>3</sup> show the ants’ response to varying environments, by having the ants travel along a V-shaped path at different angles. As shown in Fig. 2.2, bridge formation generally begins where single ants can bridge the gap (1), and widens (2) and (3), as time continues and more ants join the bridge. If we assume ants will travel the shortest path, more ants will travel along wider side (closer to the bottom in Fig. 2.2). As ants





**Figure 2.2:** Bridge experiment with *Eciton hamatum*. Bridge formation starts where ants can easily cross and then migrates upwards<sup>3</sup>.

react to local traffic flow, more ants will join the bridge at the bottom, wider side. Ants towards the top, narrow portion of the V will slowly start to leave as fewer and fewer ants travel along that path. This causes the bridge to move slowly down the V as observed. We hypothesize that this is also true for natural bridges such as the bridge in Fig. 2.1f. For larger angles, the cost of adding to the bridge is greater, so the bridge may stay closer to the starting point than for more acute angles. At very high traffic levels, such as occur during the migration of the bivouac, the ants may form much larger bridges (again, such as Fig. 2.1f) while the bridges formed during foraging are generally smaller (Fig. 2.1e).

## 2.2 SELF-ASSEMBLY IN ROBOTICS

Self-assembly in robotics could have many of the same benefits found in natural systems. Robots in a group could combine in order to overcome obstacles or perform otherwise impossible tasks, e.g. reaching or retrieving an object that would normally be out of reach or forming protective and/or escape structures like the fire ant rafts. These features could be especially useful when exploring unknown, dangerous or rapidly changing environments. The use of robots to

form temporary structures also allows for the building blocks of these structures to be completely reusable. As one structure becomes unnecessary, robots disassemble and can join a new structure or perform other tasks. Assuming no robot fails, self-assembled structures leave no trace on the environment after being disassembled.

Previous work on self assembling robotic systems has largely drawn inspiration from cells and molecules rather than ants or other insects, and has focused on the creation of predetermined, lattice-based structures<sup>18</sup>. Generally, robot modules latch onto one another at fixed docking sites. A blueprint of the desired target structure is encoded in the form of a map or a set of rules and programmed on the robotic modules, guiding the assembly process as DNA guides the assembly of an organism from cells. Self assembly in this case also allows robots to adapt to new situations and serve as reusable building blocks, but generally requires a centralized controller or outside user to provide the plan.

One popular area of research, inspired by the idea of transformable robots changing shapes to accomplish different tasks, focuses on modular and re-configurable robots. Over the past 20 years, a multitude of modular robots have been designed, modeled, and tested. Many have been in the shapes of cubes<sup>37,38,39,40,41,33,42,35,43,18</sup>, with a few exceptions<sup>44,32</sup>. These robots generally reconfigure either by sliding along other robots or pivoting around them; sliding schemes such as<sup>40</sup> or<sup>37</sup> as yet can only be applied to 2D structures, while many pivoting systems work in 3D<sup>18</sup>. Reconfigurable robots are generally structured into either chains or lattices<sup>18</sup> and are often limited in their ability to move independently, without being attached to other modules. The idea of programmable matter, where smart building blocks or particles can form and reshape into new tools on demand, has inspired other modules; systems like the programmable parts test bed by Klavins et al. or, more recently, Lily by Haghighat et al., have no mobility and



**Figure 2.3:** Examples of self-assembly in robotics. **Modular robots:** A) ATRON<sup>32</sup> B) M-TRAN III<sup>33</sup> C) Pebbles<sup>19,34</sup> D) SMORES<sup>22</sup> E) M-Blocks<sup>17</sup> F) Soft Modular Cubes<sup>24</sup>. **Mobile Swarms:** G) Kilobots<sup>6</sup> H) Robotic boats<sup>16,35</sup> I) Tank Robots<sup>36</sup>. **Adaptive Assembly:** J) Swarmbots<sup>12</sup> K) Slimebot<sup>14</sup>

depend on their environment to cause them to self-assemble stochastically<sup>20,21</sup>. The Pebbles system<sup>19,34</sup>, is likewise immobile and uses a subtractive method to form structures: the environment brings modules together, modules communicate between themselves, and, based on local rules, decide whether to stay attached or detached. Excess modules are then shaken off.

Systems for programmable matter may be useful when interacting with humans or for one time deployment in specific environments which allow them to assemble. However, if robots are to explore unknown terrain or act on their own, they will need to be mobile. Two advanced modular systems, SMORES<sup>39,22</sup>, and M-Blocks<sup>35,17</sup> use agents that are autonomous and mobile on their own, but can also self-assemble with other robots. The SMORES system consists essentially of wheeled cubes whereas M-Blocks use the momentum from a flywheel to propel themselves forward. SMORES in particular brings us closer to the idea of a re-configurable swarm; using a small team of regular SMORES in combination with a specialized camera unit, the group is able to choose its shape from a set library and reconfigure to achieve different goals<sup>22</sup>. Only these two systems and Pebbles<sup>19</sup> have been able to create structures in 3D.

In addition to the extensive work on modular robots, collectives of mobile robots also form self-assembled structures, though so far all in 2D. Kilobots have formed a variety of shapes with up to a thousand robots, with a additive self-assembly algorithm<sup>6</sup> and then again through subtractive methods<sup>7</sup>. Modular boats have also been used to form bridges over water<sup>45,16</sup>, and tank robots on land to form a vertical pyramid<sup>36</sup>.

The above robotic systems use rigid agents to create pre-determined, precise shapes. Except for the Kilobots<sup>6,7</sup> and the tank robots in Cucu et al.<sup>36</sup>, which do not physically connect and instead use stigmergy and communication to align, robots have fixed docking points which limit shape creation to lattice or chain-based shapes. In contrast, army ants have flexible bodies and

can attach to each other at any point. Their self-assembled structures can conform to many environments and dynamically adapt their shape, size, and structure to the changing needs of the colony. Few examples of this sort of adaptive or dynamic structure-building exist in robotics. In collective construction, where groups of robots build structures from outside materials (i.e. not their own bodies), TERMES robots have created staircases<sup>9</sup>, and Melenbrink et al. simulates the construction of truss based cantilevers over gaps - robots react to the forces sensed at each joint and place material accordingly. These are adaptive structures, though still lattice-based. In the realm of self-assembly, Swarmbots<sup>12,13</sup> create foraging paths and adaptive pulling chain structures for rough terrain and collective transport, inspired by the behavior of weaver ants. Slimebots<sup>14</sup> are inspired by slime molds and adaptively reconfigure around obstacles while moving on a flat surface. These systems move us forward towards ant-like structures, but are still partially user defined; the size of the structure is defined by the number of robots in the experiment rather than the needs of a specific goal, and are limited to mostly flat terrains. A newer system for self-assembly, FireAnt<sup>46</sup>, has been developed by collaborators Mike Rubenstein and Petras Swissler, inspired in part by our simulated robot model (Chapter 4). It can attach to other robots at any point and climb over other robots in 2D<sup>46</sup>. This platform has a lot of potential, but as of yet is limited to one active robot, which can climb over surfaces composed of other robots, but cannot form new structures.

So far, all of the systems discussed in this section are completely rigid. Indeed, while examples of soft collectives, including soft self-assembling collectives, abound in nature, all of the systems above, from modular robots to adaptive swarms consist of completely rigid robots. Soft robots have shown great potential - they can be incredibly robust and durable<sup>25</sup> - but soft robot swarms, whether for self-assembly or any other purpose, to our knowledge, have not yet

been developed, possibly due to the reliance of many soft robots on pneumatic air supplies. A few researchers have used soft robot modules as the building blocks for larger systems. For example, Cheney et. al<sup>23</sup> evolved soft robots in simulation from cubic modules. Soft Modular Cubes in Vergara et al.<sup>24</sup> demonstrate many interesting cellular behaviors including invagination and migration using a centralized controller and air supply. C-balls<sup>47</sup> consist of a soft ball module that moves via a magnet system attached to a passive module. This is an interesting alternative to pneumatic control, but only limited motion can be achieved with 2-3 balls. As with the Soft Modular Cubes, human researchers must pre-assemble the modules, and it is unclear how this system would generalize.

Taking inspiration from army ants, where soft-bodied, autonomous, individuals create non-lattice structures, the *Eciton robotica* system uses a flexible soft-bodied robotic module which is designed to follow simple local rules to form bridge structures that adapt to the environment and traffic conditions. To design the *E. robotica* modules, I looked for more inspiration in climbing robots, detailed in the following section.

### 2.3 CLIMBING ROBOTS

Like many of the self-assembling systems described in the previous section, including modular and re-configurable systems, self-assembly modules for adaptive structures must be able to climb both over other robots and over any self-assembled structures. Climbing robots share some of the same requirements - namely they must climb over both obstacles and steep slopes - and do so without the rigid alignment requirements of most modular robots.

Climbing robots have a large array of potential applications<sup>48</sup>, including maintenance activities in and outside buildings, monitoring and inspection, and search and rescue. They provide

access to spaces where it could be dangerous or difficult for humans to operate, whether at high altitudes, in fragile environments, or within cramped pipes.

A wide range of climbing robots has been developed, which can generally be divided into (1) robots designed to handle rough terrain and (2) robots to scale steep and/or vertical surfaces. In the first category lie robots such as Shrimp<sup>49</sup>, Mobit<sup>50</sup>, Whegs<sup>TM</sup> and Mini-Whegs<sup>TM</sup><sup>51,52</sup>, and X-Rhex and RHex<sup>53,54</sup>. These robots are highly capable - the latest versions of Rhex can climb over obstacles in the same size range as its own body, and even perform jumps to get over larger barriers - but, like most mobile vehicles, are limited to mostly flat or angled terrain and are mechanically complex. With the possible exception of Mini-Whegs<sup>TM</sup> or a highly simplified version of RHex, these robots would be difficult to scale for swarm applications, and would likely get tangled when climbing over each other. Any attachment system in these robots would likely need to be separated from the locomotion system.

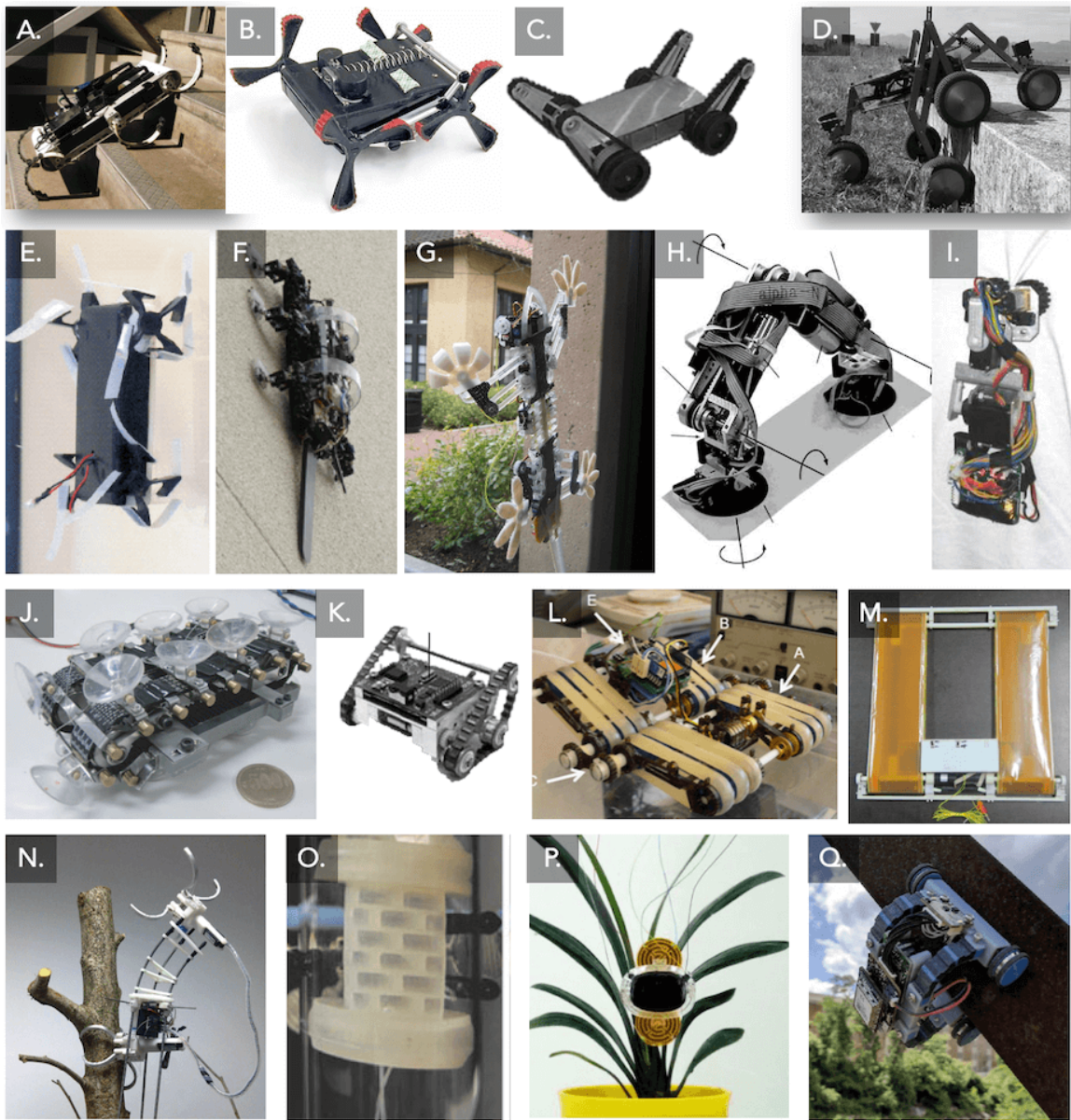
The second category of robots focus on climbing over slopes, and have mostly focused on vertical climbing or climbing ramps of increasing steepness<sup>55,56,57,58,59,60,61</sup>. Some robots are able to transition between surfaces<sup>62</sup>, climb upside down, or both<sup>63,64</sup>. The tank-like robot by Seo et al.<sup>65</sup> has impressive capabilities, including internal and external transitions and climbing over obstacles, but is unable to move from vertical walls to the ceiling. Biped robots such as RAMR<sup>66</sup> and W-Climbot<sup>67</sup>, from which our robot takes inspiration, are able to climb at many angles and transition between all interior angles. The flipping gaits in these two robots allows them to more easily transition between surfaces, and the joints provide degrees of freedom needed to conform to new terrain. Yet, unlike Seo's tank robot which uses a passive pin joint or any of the soft robots which are similarly passively compliant, RAMR and W-Climbot require either human tele-operation (RAMR) or complex path planning (W-Climbot) because they

must plan to exactly match the shape the terrain instead of passively conforming to it.

While RAMR and W-Climbot both use active suction to stick to the climbing surface, which generally requires a tether, an abundance of other methods have been developed by researchers for climbing robots. Some use passive suction instead of active<sup>68,69</sup>. Others use magnets<sup>64</sup>, or electroadhesion<sup>56,57</sup>. Many groups have looked to biology for inspiration. The Biomimetics and Dexterous Manipulation Lab at Stanford has done an extensive amount of research developing gecko-inspired adhesives, and recently used them to allow a tiny 9 g robot to climb vertically with a payload over 100x its bodyweight<sup>59</sup>. Earlier versions of the adhesive were also used on the Stickybot platform<sup>78,72</sup>. The adhesive performs well on relatively smooth surfaces, even allowing a human to climb on a glass wall<sup>59</sup>, but does not work well when climbing rough surfaces or surfaces with a lot of dust or dirt. When dealing with rougher and more natural surfaces, some researchers have turned to using micro-spines to attach to small asperities. RiSE<sup>71</sup>, its precursor Spinybot<sup>61</sup> and another robot T-bot<sup>62</sup>, for example, use micro-scale spines on the feet. RiSE has achieved success on many different surfaces, including brick and concrete, and can also walk well on flat surfaces, but it is fairly large and highly complex, with twelve motors in total, two for each leg. The complexity and cost of such a robot make it ill suited for swarm applications. Xu et al.<sup>79</sup> combines dry adhesives and micro spines for another capable, yet complex robot. Adapted Mini-Whegs™, from<sup>52</sup>, are much simpler and have been shown to climb all kinds of surfaces using tape, micro-spines and Velcro<sup>70</sup>. The robot is more successful without back legs; with them, the robot has difficulty transferring to difference surfaces or over obstacles, though a flexible body, as in<sup>65</sup> may improve performance.

Another design, the Treebot<sup>74</sup> climbs natural surfaces, especially trees, and relies on suture needles at the end of the grippers for gripping. Unlike the micro-designs, the needles are de-





**Figure 2.4:** Selection of Climbing Robots. **Robots for Rough Terrain:** A) X-Rhex<sup>53</sup>, B) WhegsTM<sup>52</sup>, C) Mobit<sup>50</sup>, D) Shrimp<sup>49</sup>. **Robots for vertical climbing:** E) MiniWhegsTM<sup>70</sup>, F) RiSE<sup>71</sup>, G) Stickybot<sup>72</sup>, H) Biped robot RAMR<sup>66</sup>, I) Clothbot<sup>73</sup>, J)<sup>68</sup>, K) Tripillar<sup>64</sup>, L) Modular tank robot using dry adhesive tracks<sup>65</sup>, M) Tracked Wall-Climbing robot with electrostatic adhesion<sup>56</sup>. **Flexible and soft robots:** N) Treebot<sup>74</sup>, O) Soft Tube Climbing Robot<sup>75</sup>, P) Soft Wall-climbing robot<sup>76</sup>, Q) Bridgebot<sup>77</sup>.

signed to drive into the surface, so the Treebot has difficulty climbing trees with harder bark that the needles cannot penetrate. The flexible body, however, allows it to easily transition between surfaces, from branch to branch. It uses three motors to tighten springs on the robot body, which allow it to stretch or contract in three degrees of freedom. Two linear actuators are used to open and close the grippers, which are closed in the passive state.

Prior to my development of Flippy in 2017<sup>26</sup>, almost all climbing robots had been limited to rigid materials and components, sometimes adding joints to allow transitions from one surface to another. One exception is the Treebot<sup>74</sup> with its flexible springs. Meanwhile, climbers in nature display much more flexibility. Larger climbers like squirrels have a skeleton structure, but are still made of soft tissues, and many insects such as caterpillars, slugs, and inchworms are completely soft and compliant. In the past few years, a few soft climbers have emerged, including Bridgebot<sup>77</sup>, which uses the same combination of flexible and stiff components as Flippy, but integrates them as legs with magnetic wheels to examine bridges. One soft robot<sup>75</sup> climbs inside tubes while another small soft climbing robot<sup>76</sup> uses a dielectric-elastomer actuator and two electroadhesive pads to inchworm its way up walls, though both of these are still tethered. Of these systems, only Flippy and Bridgebot are untethered and autonomous.

Inspired by soft climbers in nature, I designed our first robot module, Flippy, a small biped robot with a soft, flexible body and grippers on each end. The robot was equipped with only limited sensing and a simple control algorithm, but thanks to its compliant body and flipping gait, Flippy was able to autonomously climb up and down surfaces held at any angles relative to gravity, including vertically and upside down, and could transition between interior planes in different orientations. The design of the Flippy robot and the current *Eciton robotica* modules are described in the following chapter.

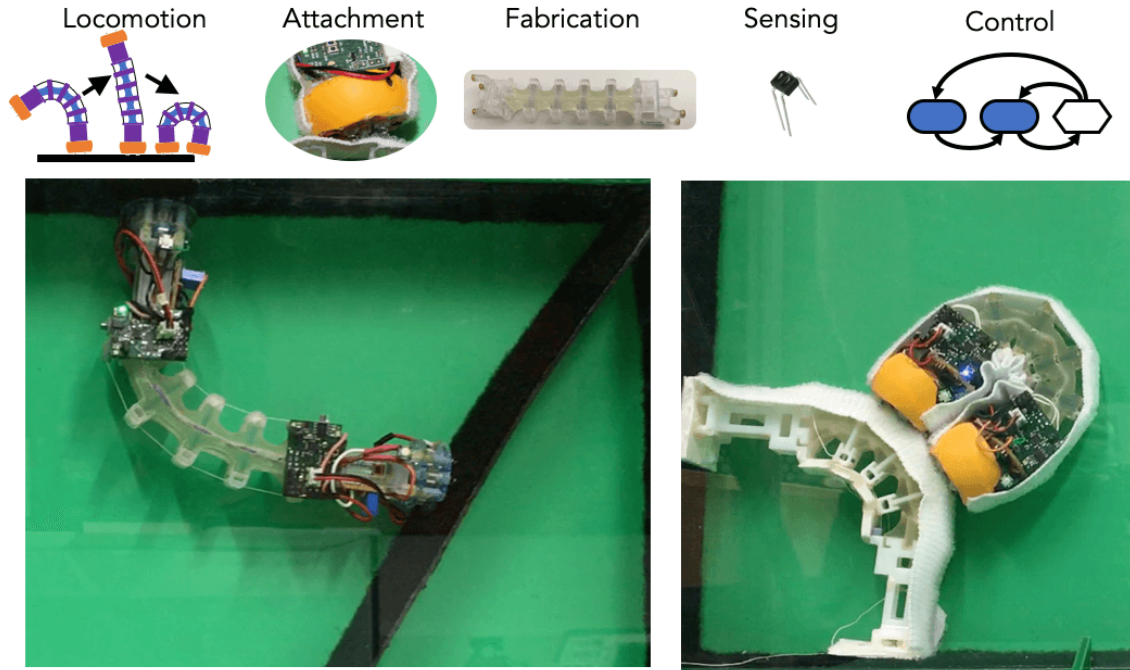
## Chapter 3

# Robot Hardware Module Design:

## Flippy and *Eciton robotica*

### 3.1 OVERVIEW

The design of any platform for collective systems must start with the individual modules. In this project, our objective was to create a module that would be capable of self-assembling army-ant inspired amorphous and adaptive structures. Figure 3.1 gives an overview of the main challenges including locomotion and climbing, attachment and detachment, fabrication, sensing and control. In response to these challenges, I designed a new robot: a soft, flexible biped



**Figure 3.1:** Design challenges for the robot hardware modules and their implementation in Flippy (bottom left) and *E. robotica* (bottom right)

that flips in order to move and climb over other robots. The flexible body allows the robot to conform to many different terrains, including other robots. The novel gripping mechanism allows the robot to attach to other robots at any point along their bodies. The first robot module design consists of a climbing robot, nicknamed Flippy, introduced in 2017 at the International Conference on Intelligent Robots and Systems (IROS)<sup>26</sup>. To my knowledge Flippy was one of the first untethered soft, flexible climbing robots and was able to climb vertically, upside down and transition between surfaces completely autonomously. This module was adapted for the multi-robot system, published in 2020 at the International Conference on Robotics and Automation (ICRA). *E. robotica* is also the first soft modules for self-assembly in our knowledge and the first to achieve self-assembly of adaptive structures in a vertical plane.

This chapter starts with general principles of the design (design goals, locomotion, and attachment mechanism) and then moves to the implementation of these principles on real robots: Flippy and *E. robotica*.

**3.2 Design Objectives** describes the design goals of the robot modules.

**3.3 Locomotion** describes the flipping gait and design of the robot body.

**3.4 Gripper Mechanism and Design** describes design process of developing the novel corkscrew gripper. I start with concept selection and testing of several proof-of-concept designs, discuss modeling and optimization of the corkscrew gripper, and finish with final integration and verification.

**3.5 Flippy: A Climbing Robot** describes the integration of the body and gripper designs in the first robot module, Flippy, including fabrication, electronics, control, and climbing experiments.

**3.6 Eciton robotica** discusses the necessary changes to go from a climbing robot to modules for self-assembly, including incorporating the Velcro covering and modifications for climbing over other robots.

## 3.2 DESIGN OBJECTIVES

To be able to self assemble into ant-like structures, at minimum a robot must be able to:

1. **Climb over each other and over any self-assembled structures.** For structures to grow and adapt to various conditions, robots must be able to climb over any structure they create, which may include **steep or even inverted slopes**. Self-assembled structures composed of robots may also make for **bumpy and uneven terrain**.

2. **Attach and detach to a fellow robots.** For amorphous, ant-like structures, robots should be able to attach **at any point on another robot's body**.

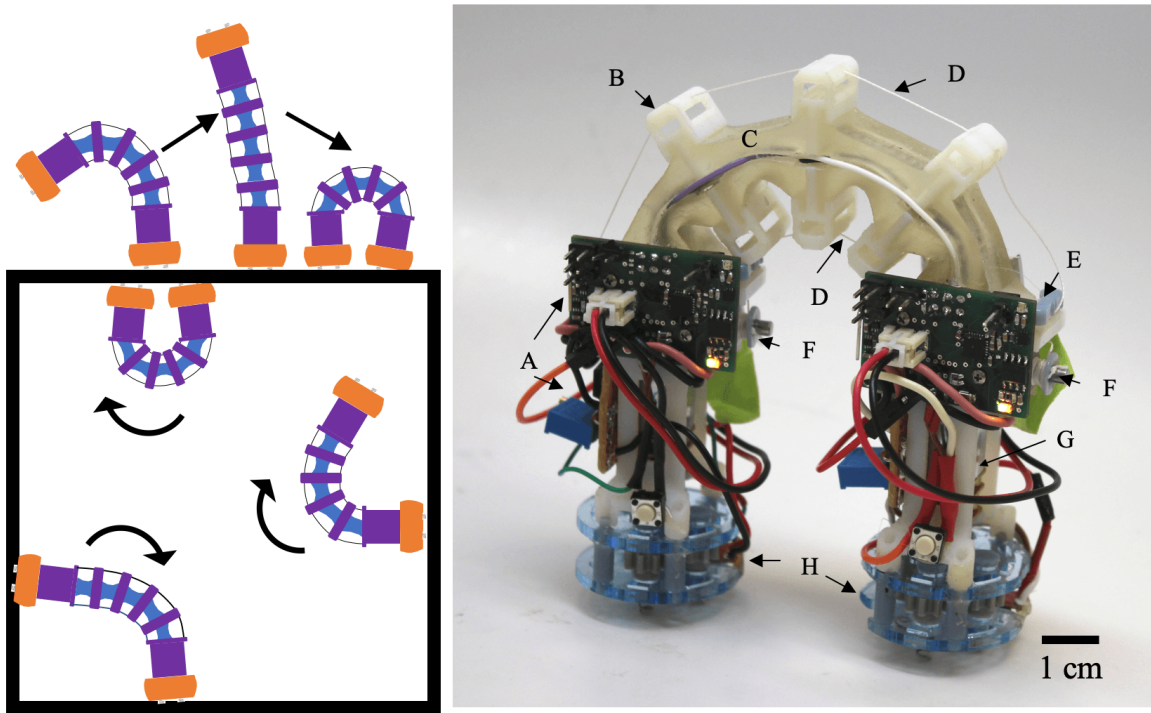
In addition, as a collective mobile swarm, robots needed to be **cheap and easy to manufacture, autonomous, and untethered**. Priority was given to designs that are simple but reliable and can recover from errors. For feasibility of experiments, we aimed to minimize size, while keeping the robot easy to handle.

### 3.3 LOCOMOTION

The Flippy robot, shown in Fig. 3.2 on the following page, is a small biped robot which utilizes a flexible body and flipping gait to climb and transition between surfaces. The robot alternates gripping and flipping to move forward and climb. The body is composed of alternating 3D printed rigid (B) and flexible (C) segments. Two DC motors (G) on either side of the robot wind and unwind spools (F) of cable (D) to control the bending of the robot and execute the flipping gait.

Though flipping mostly calls to mind human gymnasts, a few animals also use similar gaits: some spiders, caterpillars and other animals use wheel like motions to escape predators<sup>80</sup>, and the adult stromatopod *Nannosquilla decemspinosa* flips its body when stuck out of water<sup>81</sup>. Flipping has a few drawbacks: it requires a gripper that can counterbalance the moment arm of the fully extended robot body length, and it is slower and less efficient than wheels, legged or even inchworm gaits. While these gaits are faster over flat surfaces, they may struggle or require more complex control when presented with obstacles or transitions between surfaces. Flipping, on the other hand, allows robots to easily transition between surfaces at different angles, as demonstrated by two previous rigid biped robots<sup>66,67</sup>. Similarly, most re-configurable robots





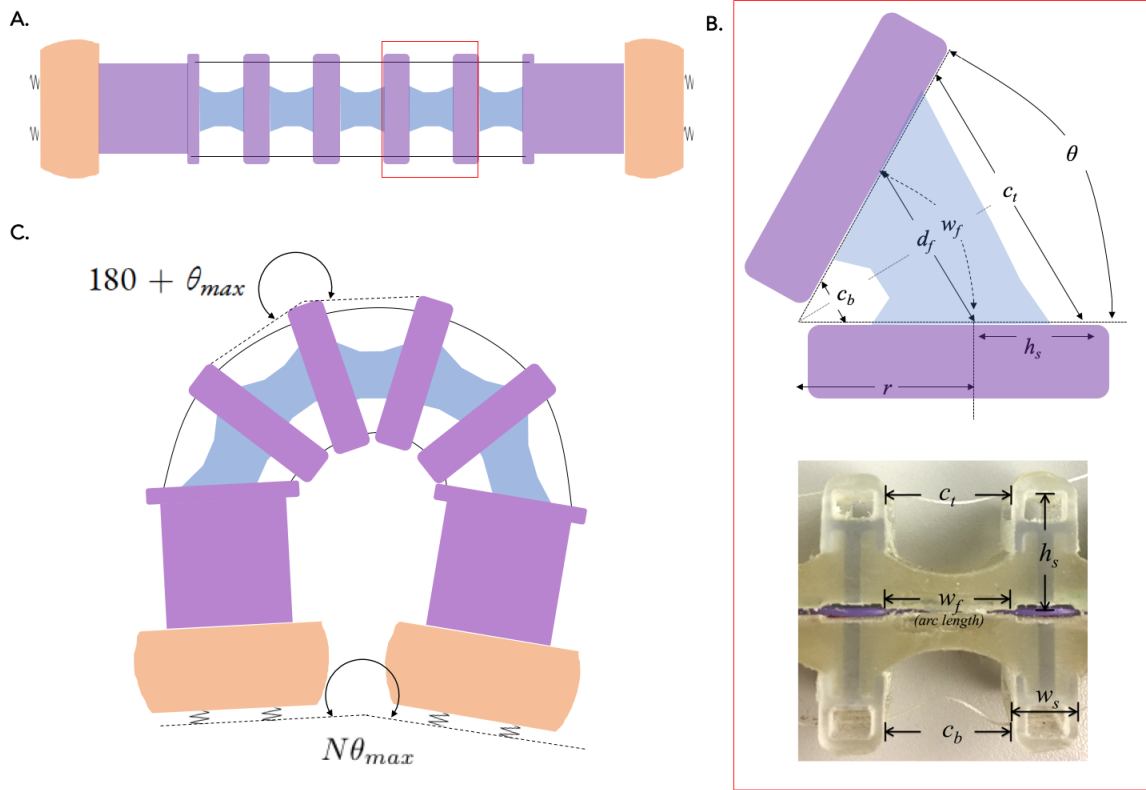
**Figure 3.2:** Left: The flipping motion of a single robot as it moves on a flat plane (top) or around a box (bottom). Right: The Flippy Robot. A) Printed circuit board and additional board for IR sensor circuit B) Stiff body part C) Flexible body part with embedded wire to connect between boards D) Cable E) Tension sensor: a pin joint link which pushes a switch on the PCB when the cable is in tension F) Winding spool and motor G) Gripper motor H) Corkscrew gripper

which climb on top of one another such as the M-blocks<sup>35</sup> operate on a similar principle, using the pivoting cube model. This indicates that flipping may be a useful gait in self-assembly as well as climbing.

In the Flippy robot, the use of flipping in a compliant body eliminates the need for any sort of complex control. The gait essentially acts as a search pattern for the closest available surface, as shown in the top left of Fig. 3.2. Once the robot finds a surface, the flexible body allows the robot gripper to conform to the surface angle without the use of a human operator or complex control system, allowing it to easily transition over surfaces as shown in the bottom left panel of the Figure.

### 3.3.1 BODY DESIGN

The body of the robot is modeled after shape deposit manufacturing finger designs used for robotic hands and graspers<sup>82,83</sup>. Fig. 3.2 shows the full robot. Cables (D) on the top and bottom are drawn through stiff sections (B) of the body and tightened or loosened by motors (G), causing the flexible portions (C) of the body to bend. Tightening one cable while loosening the other creates a flipping motion. Grippers at both ends (H) attach to surfaces at any orientation and are explained in further detail in Section 3.4.



**Figure 3.3:** A) Outline of the robot B) Close up of a robot segment. Top shows bending due to the change in cable length  $c_b$ ; bottom shows the shape and dimensions on the real robot. C) A fully bent robot where the total bending angle is bending of each segment  $\theta$  times the number of segments  $N$ . The outer curvature is  $180 + \theta$



To design the body and locomotion of the robot, I created a simple, geometric model. I assumed that the winding cables move at constant speed so that the system is always at equilibrium. For simplicity, I ignored the viscoelastic behavior of the flexible material and assumed negligible friction i.e. the cable will shorten equally in all segments. Here segments are identical; due to the flipping gait of the robot, the body should be at least symmetric so this was a reasonable assumption. The model could also be expanded to account for variation in segment lengths. In the physical robot, curved flexible segments (as shown in Fig. 3.3a) encouraged bending at the center of the flexible sections, which made it easier to model and kept the physical robot and model more consistent with each other. Using these assumptions, I reduced the scope of the model to one segment only, used this to inform the design of the robot, and then modeled the flipping gait.

A close up of one bent segment is shown in Fig. 3.3b. Here  $w_f$  is the width at the center of the flexible component,  $c_b$  the bottom cable length,  $c_t$  the top cable length, and  $h_s$  the height of the stiff component from the cable to the center.  $r$  represents the distance from the center point of the stiff segments to the point of the extended triangle (also the center of the curve of arc-length  $w_f$ ). The bending angle  $\theta$  is the angle between the two stiff segments. In Fig. 3.3a or b - bottom, when the robot is not bending and the stiff segments are parallel, this is simply zero.  $w_f$  will become an arc length as the flexible segment bends; I assume for simplicity that the magnitude of the length will remain constant i.e. that elongation and compression will occur above and below the center line, but the center line arc length will remain constant. Maximum bending will occur when the two stiff components collide, i.e. when  $r$  is equal to  $h_s$ . Thus the

maximum angle in radians is simply

$$\theta_{max} = w_f/h_s \quad (3.1)$$

This can be used to determine  $h_s$  and  $w_f$  given a desired angle. The total maximum bending angle of the robot will be the sum of the maximum bending angle of each segment or  $\sum \theta_{max}$ .

### BODY DESIGN CONSIDERATIONS

The following design considerations were crucial for choosing the desired angle and dimensions:

- **Total bending angle:** To navigate flat surfaces, the robot must bend at least  $180^\circ$  from its unbent position, both shown in Fig. 3.3c. Climbing vertical walls requires an extra safety factor, as the robot may tilt away from the wall (discussed later, Fig. 3.10). Therefore, *the robot should have a total maximum bending angle of at least  $200^\circ$ .*
- **Bending angle and dimensions of segments:**
  - **Height of the rigid sections  $h_s$ :** As shown in 3.1, to get large bending angles in a single segment, we need a small  $h_s$  relative to  $w_f$ . However, larger  $h_s$  values will require less winding cable force, as can be seen in Fig. 3.6b and discussed in the following subsection on actuation. *Adding segments allows us to have a smaller  $\theta_{max}$  for each segment while maximizing the overall bending angle, thus allowing for larger values of  $h_s$  relative to  $w_f$ .*
  - **Width of the flexible sections  $w_f$ :** Although we need a longer  $w_f$  relative to  $h_s$  for more bending, *longer flexible parts add degrees of freedom and reduce the pre-*

*dictability of the robot.* The stiff portions of the robot serve both as a guide for the cable and as physical constraints.

- **The width of the stiff sections  $w_s$ :** The width  $w_s$  affects the length of the robot but not the bending angle and should be minimized. However, it must be wide enough for a continuous stiff section and flexible portions as shown in Fig. 3.3b to prevent delamination of the flexible material from the rigid surface. In addition, these sections provide spacing which may be necessary to prevent the collision of the grippers when the body is fully bent.
- **Height of the flexible sections  $h_f$  and overall thickness:** These do not materially affect the bending geometry but do impact the material properties and dynamics of the system (i.e. the force required to bend).
- **Number of segments:** The model uses similar triangles and cannot be used for segments with bending angles of  $180^\circ$  or more. Therefore, *the model assumes that the robot will consist of at least two or more flexible segments. The penalty for adding segments is small but increases with the number of segments while the benefits are high, but decrease with the number of segments as  $1/N$  where  $N$  is the number of segments.* This is due to several factors including:
  - **Length:** Adding segments increases the length of the robot and its the moment arm. Each segment added consists of not only the flexible portion but a stiff portion which does not contribute to bending. However, the flexible body accounts for only a portion of the length and mass of the robot; the mass of the actuators and grippers will remain constant for a given actuator and gripper. Since these

constants account for a larger portion of the mass and length (approximately two-thirds of the weight and almost half the length in the Flippy and *E. robotica* robots), the penalty for adding segments is relatively small to start and remains relatively linear, increasing slightly with additional segments.

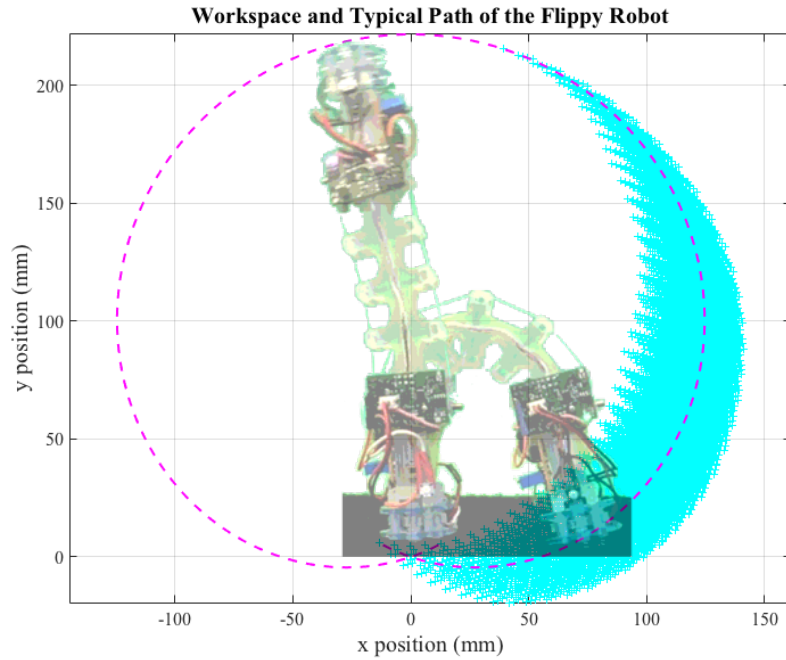
- **Cable friction:** Adding segments reduces cable friction, as it prevents the cable from wrapping around sharp angles. The outer angle is directly proportional to  $\theta$ , which decreases with  $1/N$ .
- **Bending angle:** Overall, the bending requirements for each segment decrease with  $1/N$ , including cable friction, the outer angle (which the robot must climb over if it is to climb over other robots) and the ratio of  $w_f/h_s$ , which is reflected in the cable tension and torque requirements for actuation.
- **Scaling:** The robot was chosen to be approximately the size of soda can, on the scale of centimeters. Several factors went into sizing the robot including:
  - **Feasibility of experiments:** For the purposes of testing in lab space, robot size was minimized. Testing space for the robots scales with  $L^2$ : a 1 cm<sup>2</sup> robot may require a 100 cm<sup>2</sup> space, while the same experiment with a 5 cm robot will require 2500 cm<sup>2</sup>.
  - **Ease of handling and manufacturing:** Humans excel at handling objects within a certain size range; the robot is within the range of most hand tools. Since the robot is 3D printed for ease of manufacturing, it is also constrained by the total printing capacity and printing time, which scales with the volume of the robot  $L^3$ .

- **Scaling laws:** The moment arm of the robot scales with  $L^4$ , and mass with  $L^3$ .  
Meanwhile most attachment schemes scale with pressure and area,  $L^2$ , making it to our advantage to minimize the overall scale of the robot.
- **Availability of cheap, off-the shelf parts:** The robots are limited by the size of the actuators, in this case, the motors which wind the cables, the selection of which is discussed later in this section.

#### FINAL BODY SPECIFICATIONS

The final Flippy robot consisted of four flexible segments, each with a  $\theta_{max}$  of  $57^\circ$  for a total of  $228^\circ$  of bending. For the dimensions of the Flippy robot,  $h_s$  was 15.2 mm, the minimum height to accommodate the motors. From the chosen angle and 3.1,  $w_f$  was also 15.2 mm. To minimize the length (and the moment arm) of the robot, the width of the stiff components,  $w_s$ , was made as small as possible, while allowing for space between the two grippers.

The material of the flexible sections was mostly chosen for ease of prototyping; new ideas could be quickly 3D printed. While the material (TangoPlus) can be combined with a rigid material to increase shore hardness, testing indicated these were more prone to breakages and cracks. I chose to keep the flexible portions simply as TangoPlus, and the thickness  $h_f$  was chosen experimentally as the minimum thickness capable of supporting the robot moment arm, i.e. the robot body remained straight when held at one end perpendicular to gravity. The spring force of the body thus both helps the body to move in the first half of a flip, but does less to impede movement in the second half. This worked well for our purposes, though more modeling and testing of different materials could be used to optimize the robot in the future.



**Figure 3.4:** Flipping trajectory of the robot. The dotted line shows a typical flip, assuming equal bending in all sections. The blue markers show the workspace of the robot.

## MODELING LOCOMOTION

Using the chosen dimensions and assuming centered bending (again, encouraged by the curvature of the flexible segments), the robot was modeled as a four joint manipulator to calculate the trajectory shown in Fig. 3.4. The center of the flexible segments were treated as revolute joints. The Denavit-Hartenberg convention was used to calculate the position of the moving gripper relative to the attached gripper. The dotted line of Fig. 3.4 shows the trajectory of the robot, assuming that the cable length is the same in all segments. The blue markers show the workspace, removing this assumption and sampling  $5^\circ$  increments of each angle. The overlap of points shows that even in this simple model, in almost all x-y positions (the gripper is h, the gripper has many possible orientations; the flexible body allows it to conform to many surface

angles. Although the flexible joints in the following model are simple bending joints, each section also has approximately  $\pm 3$  mm of translation, can be compressed, and of course can bend anywhere along the flexible section, allowing even more possible contact orientations.

The plotted workspace shows that the moving gripper has a range up to  $228^\circ$  and may occupy the same space, essentially, as the attached gripper. This is true if we allow the robot to move out of plane and place one gripper in front of the other. However, if, as in the testing described in the following sections, the robot is restricted to a straight track, the workspace is slightly smaller as the grippers cannot physically occupy the same space, limiting the maximum angle to  $210^\circ$ . In addition, due to friction, the cable length in sections closer to the motor will shorten more quickly than in the farther sections, giving us a slightly different trajectory than the one modeled. To account for this, I chose to focus on traveling in a single direction. If the winding motor is the attached gripper, this lack of symmetry can be helpful; the robot reaches past the predicted trajectory in this direction, an advantage in vertical climbing, though possibly a disadvantage in some transitions. Finally, though I do not consider them here, the robot's starting position and relative orientation to gravity will also effect the trajectory.

### 3.3.2 ACTUATION

The flipping motion is caused by the winding of cable spools, shown in Fig. 3.2F. I use nylon thread for the cable, which winds smoothly onto the spools more easily than fishing line or wire but has a higher yield strength and lower friction coefficient than standard cotton or polyester thread. Winding one side of the robot will cause it to bend, while the opposite side simultaneously unwinds. To sense when to unwind, I use a mechanical switch to sense tension in the cable.

## TENSION FEEDBACK

The robot uses a switch as a feedback system to sense tension in the bottom cable and decide to unwind. It is important that both cables stay relatively tight, as loose cable may get caught or tangled, causing errors and possibly even damage to the robot. Prior to implementing the tension switch system, I determined the relative winding and unwinding speeds of the motors while flipping and the dependence of theta on cable length, using the geometric model described in Section 3.3.1.

Treating  $h_s$  and  $w_f$  as constants, I calculated the dependence of  $\theta$  on the length of the bottom cable  $c_b$ , and the relation between the change in lengths of the bottom cable  $c_b$  and the top cable  $c_t$  or  $\Delta c_b / \Delta c_t$ . Using basic trigonometry

$$c_b = 2(w_f/\theta - h_s) \sin(\theta/2) \quad (3.2)$$

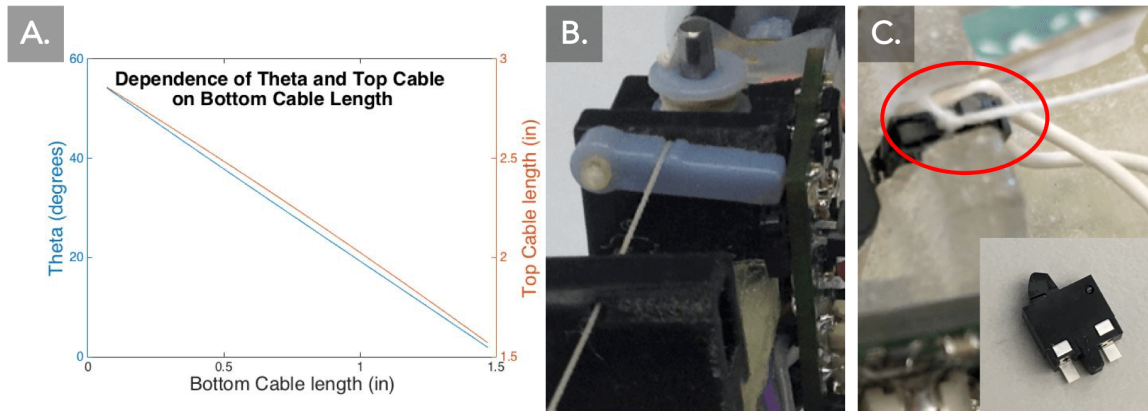
and

$$c_t/c_b = \frac{w_f/\theta + h_s}{w_f/\theta - h_s} \quad (3.3)$$

Using Matlab, I solved  $c_t$  numerically for a sample of lengths  $c_b$ . The results were plotted (see Fig. 3.5a) and linearized via the polyfit function to find  $\Delta c_b / \Delta c_t$ . The process was repeated for several values of  $h_s$  and  $w_f$ , giving us a range of values for  $\Delta c_b / \Delta c_t$  from about -0.87 to -0.95.

Winding the cables at these speed ratios would allow us to perform half a flip, from a neutral (unbent robot) to  $180^\circ$  as shown by the background robot in Fig. 3.4. The first half of the flip,  $-180^\circ$  to neutral, can be accomplished by treating the winding motor as the top cable and inverting the speed ratio. For simplicity and to prevent tangling of the cables, I instead implemented a feedback system using the tension switches shown in Fig. 3.2E and in Fig. 3.5. In





**Figure 3.5:** Tension Feedback Design. A) Dependence of bending angle and top cable length  $c_t$  on the bottom cable length  $c_b$ . While linear, the change in cable length is not simply equal and opposite, instigating the need for the feedback systems shown on the right. B) The lever design implemented on the Flippy robot C) The direct connection implemented in *E. robotica* and a close-up of the switch used in both designs.

a flip, the unwinding motor simply unwinds whenever the robot senses tension in the cable, when the cable pushes a lever into a mechanical switch.

The feedback system for the Flippy and *E. robotica* robots is shown in detail in Fig. 3.5. The lever system for the Flippy robot is shown on the left, Fig. 3.5a. The cable runs across the lever and, when in tension, will pull down slightly, and press a switch mounted on the PCB (the same as in Fig. 3.2(C) on the right). This was highly effective, but in some configurations, the cable would move away from the lever and not push it enough to create a signal. The small lever also slightly increased the length of the robot. Later *E. robotica* modules switch to a direct system, as shown in Fig. 3.5b, where the cable is tied to the switch. This method is accurate but requires careful manufacturing in order to tie the cable and not damage the delicate switch. The switch must also be glued into the 3D printed body, again without damage.

In practice, this aspect of the robot design remains one of the most fragile parts of the robot, in part because few off-the-shelf switches exist with the necessary design requirements. The tension switch needs to be millimeter scale, highly sensitive, and durable. Most switches are

designed to interact well with humans exerting high forces in compression whereas our ideal switch is highly sensitive to tension. Small strain gauges were also considered, but strain gauges in these size ranges are often brittle and easily damaged, and would require a casing or other protection.

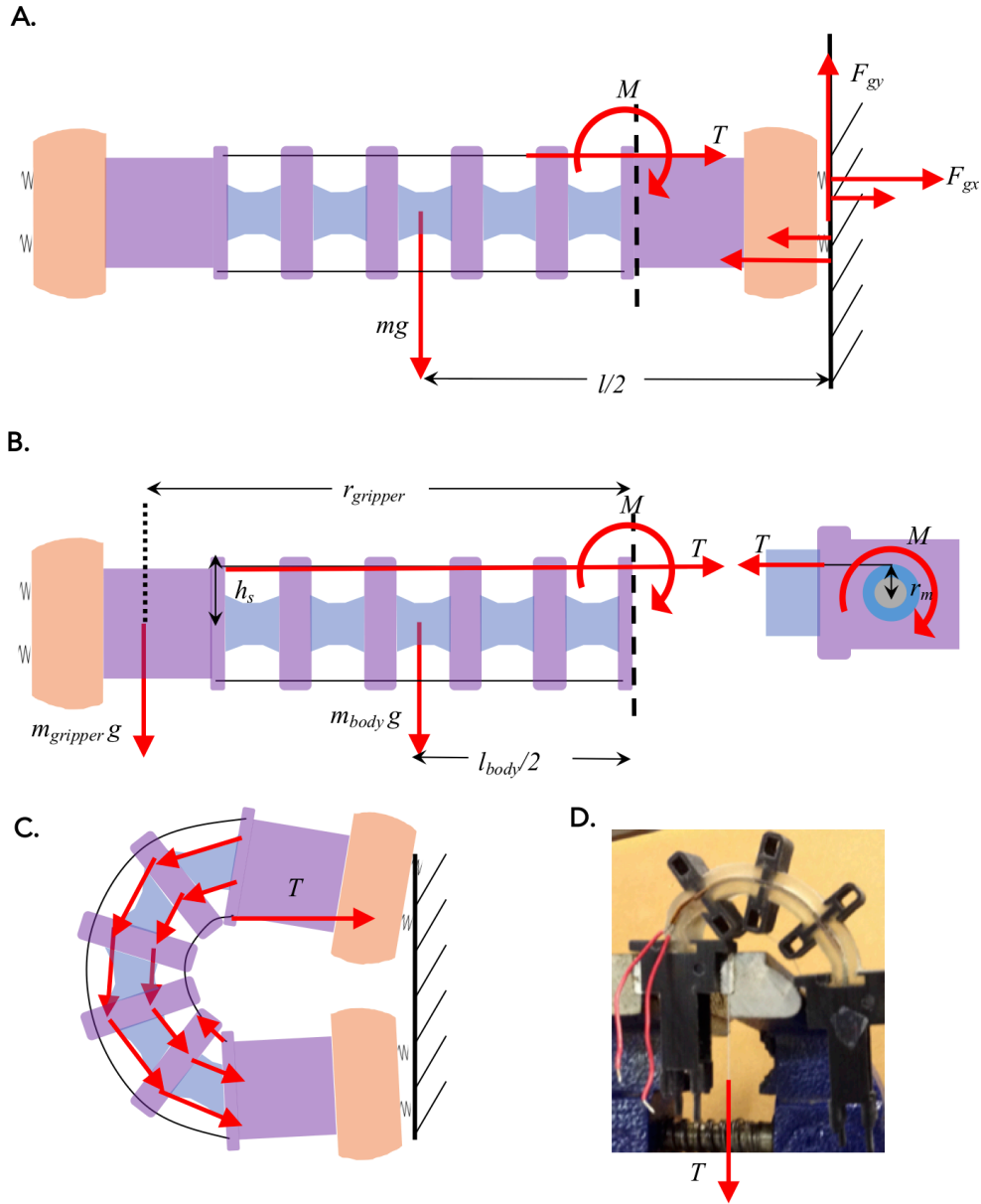
## MOTOR SELECTION

Pololu Micro Metal Gearmotors were chosen for the winding motors for their small size, price, and variety of speed and torque capacities. The necessary motor torque is equivalent to the tension  $T$  in the top cable when bending the robot multiplied by the radius of the motor shaft and spool  $r_m$ , as shown on the following page in Fig. 3.6b, right. Since

$$\tau_m = Tr_m \quad (3.4)$$

the motor torque is related proportionally to the  $r_m$  and I thus kept this dimension of the cable spool as small as possible, about 5 mm.

Fig. 3.6a gives a summary of the forces in the system: the weight of the robot, gripping force and tension caused by the winding of the cable. The gripper is in equilibrium, therefore I can treat the end segment as fixed and focus on the moment around the joining of the flexible segments to the end of the robot (shown as a dashed line). In addition, I assumed that the robot moves at a constant speed and that the system is in equilibrium, since the robot does not need to move or accelerate quickly. Thus the sum of the moments due to tension in the cable, the weight of the robot, and the elasticity of the flexible sections must equal zero. I broke the weight of the robot into the weight of the body, applied at the center of the flexible portion,



**Figure 3.6:** Free Body Diagrams for Locomotion. A) Overview of forces in the system, i.e. force from the weight of the robot, gripping force and tension caused by the winding of the cable. B) Free body diagram for calculating the bending moment of the robot from the first flexible piece. Here I assume any moments from the weight of the gripper and gripper force will cancel. The right shows a top view of the cable and torque from the motor. C) The bending of each flexible portion will create opposing forces and moment from the tension and compression in the top and bottom of each flexible piece respectively. D) Experimental determination of the tension force necessary to bend the robot  $180^\circ$

and the weight of the gripper, applied at its center.

$$\sum M = Th_s - m_bgl_b - m_ggr_g - \sum M_f = 0 \quad (3.5)$$

$T$  is the tension in the cable,  $h_s$  is the half height of the stiff body sections,  $m_b$  and  $l_b$  are the mass and length of the body respectively,  $m_g$  and  $l_g$  are the mass and length of the gripper, and  $M_f$  is the moment from the bending of the flexible sections of the robot. To calculate the necessary tension, I considered the worst case scenarios: vertical climbing with the robot fully extended Fig. 3.6b, and a fully bent robot Fig. 3.6c. Note that neither of these will happen at the same time in the real system, but summing each worst case scenario, I add a factor of safety.

The necessary bending torque was found experimentally by hanging weights from the control cable until the robot was bent  $180^\circ$  as shown in Fig. 3.6d. The robot was held vertically to minimize the effect of the weight of the robot body. This test allowed me to somewhat account for friction between the cable and the robot, which becomes significant when the robot maximally bent, and could be difficult to model. The tension was approximately 500 g, or 5 N. Since  $T$  is applied at  $h_s$ , this is about 0.076 Nm.

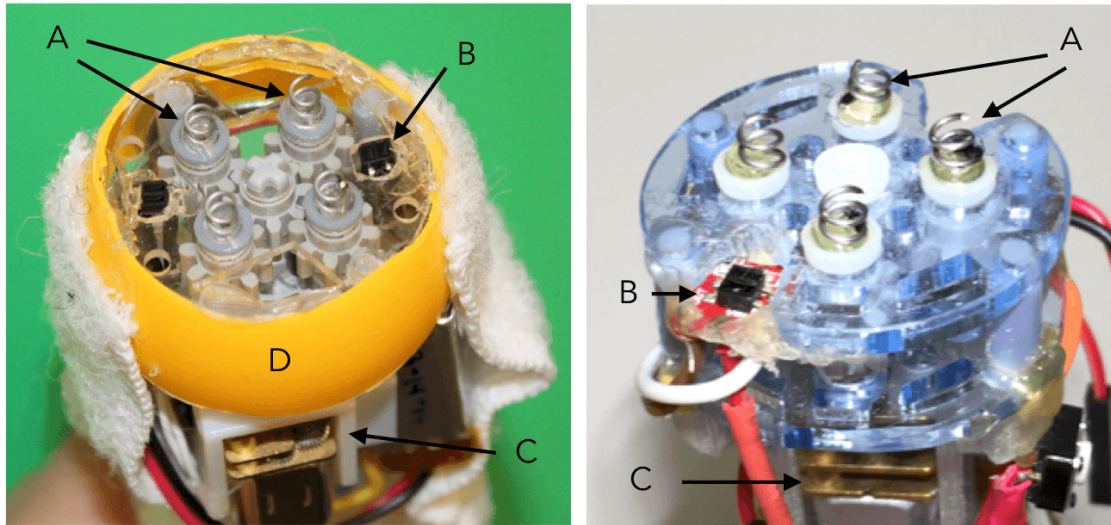
The torque applied by the moment arm of the robot could be simply approximated as  $mgl/2$ , which is 0.12 Nm, for a 200 mm long, 125 g robot. For our more complex rendering, of  $m_bgl_b + m_ggr_g$ , I get 0.0825 Nm for the same 200 mm long 125 g robot, where the body is 45 g and each gripper weighs 40 g. I used the larger value of 0.12 Nm for safety. Thus the worst case scenario is 0.076 Nm + 0.12 Nm or about 0.2 Nm. From 3.4 and 3.5 the tension in the cable,  $T$  is

$$T = 1/h_s \sum M = \tau_m/r_m \quad (3.6)$$

so  $\tau_m = r_s/h_s \sum M$  or about 0.07 Nm or about 10 oz-in. This was multiplied by a factor of safety of 3 for the motor selection. Since the robot runs at 3.7 V and not the standard 6 V used for the Pololu motors, I chose a gear ratio of 298:1, one of the highest available options, which for 6 V is rated to 56 oz-in. One large factor that these calculations overlook is the friction between the gripper and the surface. Later versions of the robot also have additional bending resistance due to the Velcro covering.

### 3.4 GRIPPER DESIGN

For the Flippy and *E. robotica* robots, I developed the novel corkscrew gripper shown in Fig. 3.7. While the gripper is most efficient in attaching to Velcro, it can attach to a number of fabric-like surfaces including foams, felts, and carpets. By covering the robots themselves in

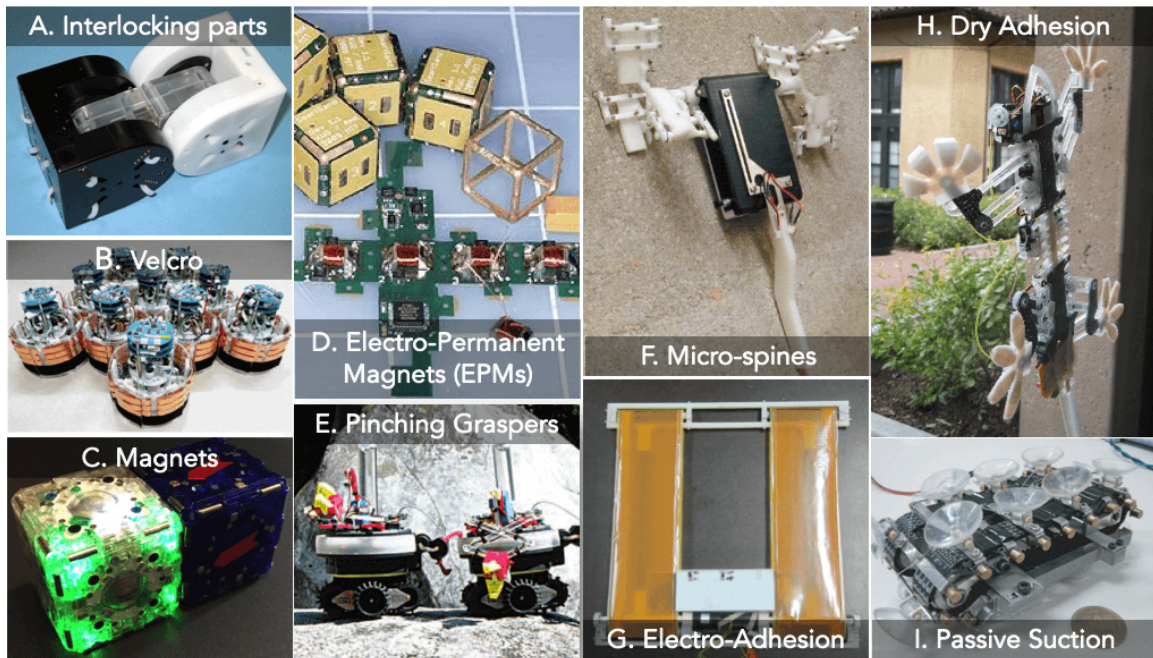


**Figure 3.7:** Corkscrew Gripper Design, as implemented in *Ecton robotica* (left) and the Flippy robot (right). Small springs form corkscrews (A) which are attached via 3D printed parts to the gear train and actuated via a DC motor (C). IR reflectance sensors (B) allow the gripper to sense nearby surfaces and attach or detach. Both gear trains are housed within two laser-cut pieces of acrylic. *E. robotica* grippers also have an orange plastic outer housing to reduce friction when sliding along surfaces.

Velcro, I created robots that attach to other robots. The gripper has several advantageous features: Attachment and detachment are simple, requiring no special alignment as long as there is a surface available. No energy is required to stay attached, and the gripper can support large forces in tension and peel.

### 3.4.1 BACKGROUND

Attachment is a critical design question for self-assembling robots. Many self-assembling robots use complex latching mechanisms<sup>32,33</sup>, magnets<sup>17,22</sup> or electro-permanent magnets<sup>19,21</sup>. Of the few examples of amorphous or adaptive self-assembly in the literature, Swarmbots<sup>12,13</sup> uses a

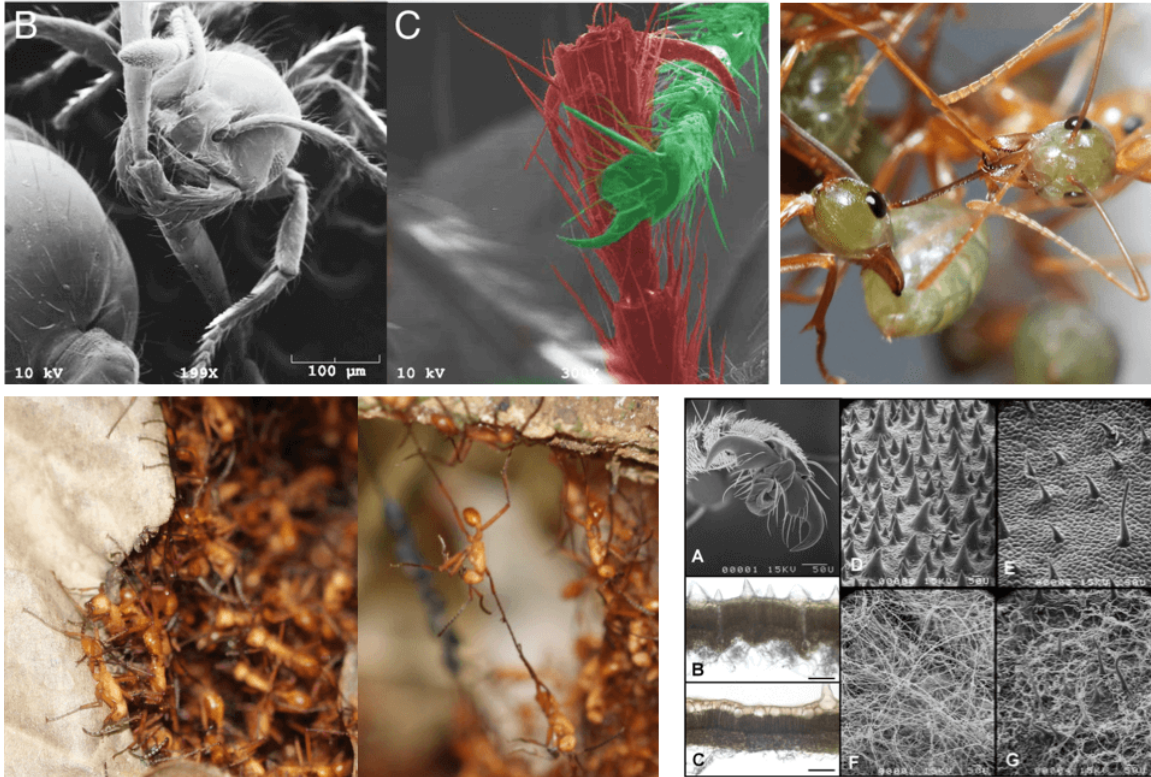


**Figure 3.8:** Attachment Mechanisms in Robots: A) Interlocking parts in modular robots - M-TRAN III<sup>33</sup>, B) Velcro for amorphous structures in Slimebot<sup>14</sup>, C) Magnets in modular robots - M-Blocks<sup>17</sup>, D) Electro-Permanent Magnets in Pebbles<sup>19</sup>, E) Pinching Graspers in Swarmbots<sup>12</sup>, F) Micro-spines<sup>70</sup>, G) Electro-Adhesion<sup>56</sup>, H) Dry Adhesion in Stickybot<sup>72</sup>, I) Passive Suction on a tank robot<sup>68</sup>

traditional claw-like grasping mechanism, and Slimebot<sup>14</sup> uses Velcro, though robots remain on the ground in this example and do not use the connection to support the weight of other robots. Climbing robots use many different mechanisms, including magnets<sup>64,84</sup>; electroadhesion<sup>57,56</sup>; wet and dry adhesion<sup>60,59,55</sup>; needles, hooks and micro-spines<sup>61,71,62,74</sup>; and pinchers and more traditional manipulators<sup>58,85</sup>. Many of these, such as electroadhesion and most adhesives, are designed to work well with robots that sit close to the wall; they are strong in shear but weak in peel. These would require significant modification to be applicable to a flipping robot which applies a large moment to the gripper. Two of the climbing bipeds<sup>66,67</sup> which inspired our robot use active suction. This allows attachment to different surfaces types and supports the required torque but requires high power consumption and is almost always tethered. The other biped, Treebot<sup>74</sup>, uses a pincher gripper with needle tips. This is closer to my design, and works well for tree trunks and branches, but may struggle with flat surfaces.

Ants, meanwhile, have a variety of methods to grasp surfaces. Weaver ants appear to mostly grasp each other with their mandibles as shown in Fig. 3.9 (Top Right), though they also stick well to surfaces with their feet, supporting weights orders of magnitude larger than themselves, e.g. in prey capture<sup>30,31</sup>. Fire ants grasp with both their tarsal hooks and mandibles as shown in Fig. 3.9 (Top Left)<sup>28</sup>. Army ants seem to mostly grasp with the tarsal hooks of their feet as shown in the bottom left of the figure. Tarsal hooks are also used for purposes beyond self assembly: *Azteca andreae*, an arboreal species, capture large prey thousands of times the mass of a single ant and use their tarsal hooks to stick to the downy sides of leaves like Velcro<sup>86</sup>.





**Figure 3.9:** Top Left: SEM images of fire ants grasping with mandibles and tarsal hooks. Image by<sup>28</sup>. Top Right: Weaver ants grasping with mandibles ©Chris Reid. Bottom Left: *Eciton hamatum* ants in a bivouac grasping with tarsal hooks. Bottom Right: SEM image of the tarsal hooks of *Azteca andreae* (A), double sided leaves (B, C), and close up images of the top and bottom side. The bottom sides have a fuzzy, velcro-like surface (D-G). Image by<sup>86</sup>.

### 3.4.2 GRIPPER DESIGN REQUIREMENTS FOR SELF-ASSEMBLY

As discussed above, while many designs and methods have been developed for attachment in both climbing robots and robots for self-assembly, gripping and attachment remain challenges in both fields. To this, amorphous self-assembly, where robots actively climb over each other and attach at any point, adds additional challenges. The following were the key design requirements for a gripper to achieve ant-like self-assembly:

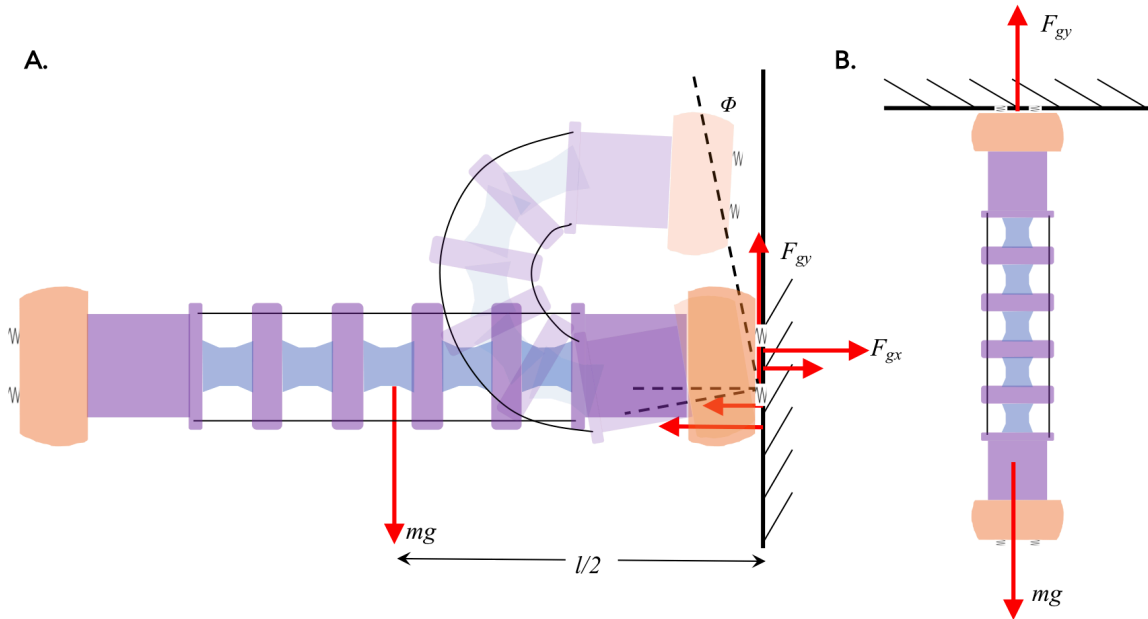


1. **Robots must be able to attach to each other at any point** to allow for adaptability and a variety of structures. To achieve this, I allowed for the possibility of coatings and coverings for the body of the robot. However, attachment mechanisms which required complex alignment, or specific latching sites were not considered.
2. **Once attached, robots should be able to stay attached without further energy output.** Bridges and structures should be able to stay in place for substantial periods of time. Once they have joined the structure, robots should be able to essentially go to sleep, only using energy to sense if they are still needed within the structure, i.e. if they are being walked over.
3. **Attachment mechanisms should be strong in tension, sheer, and peel strength.** In natural structures, a single ant may hold many of its fellow ants. To allow for the formation of chains and cantilevers, the grippers should ideally be able to hold more than one robot, whether the robot is upside down and the gripper is in tension (e.g. Fig. 3.10b), or when the robot is climbing vertically and the gripper is subjected to sheer and peel (Fig. 3.10a). Due to the flipping gait, **the gripper must also be able to support the moment arm of the extended robot and should not tilt far away from a vertical surface while the robot is climbing**, ideally not more than about  $10^\circ$  (Fig. 3.10). *Thus the torque to bending ratio is also a key factor.*
4. **Strength to size and strength to weight ratios should be maximized.** The strength to weight ratio is important in any climbing robot. Many gripping mechanisms depend on large surface areas in order to stick to surfaces. Since the climbing surface for the *E. robotica* system is other robots, increasing the surface area of the gripper also increases

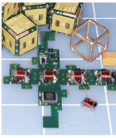

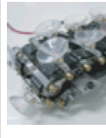
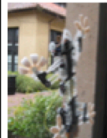


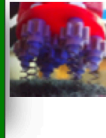

the size (and likely mass) of the robot, making it again more difficult to climb. Since the objective is to test many robots in a lab setting, small robots are advantageous for practical purposes as well.

5. Finally, I prioritize *simple designs*. **Attachment and detachment should be easy and reliable.** Grippers should be **quick and easy to manufacture**, with mostly cheap, off-the shelf components.

Unlike flipping locomotion, which stood out in its simplicity and potential for self-assembly, choosing a gripping mechanism that could meet all these challenges was less clear. I considered and tested several different options, discussed in the next section.



**Figure 3.10:** Gripper Design Requirements and free body diagram. The gripper must be able to support the moment arm of the extended robot and should not tilt far from the surface to allow the robot to climb vertically.

Design Considerations	Multiplier	 Electro-Permanent Magnets	 Permanent Magnets	 Passive Suction	 Directional Adhesives	 Pinchers	 Velcro Legs with Hook and Loop	 Corkscrew Grippers	 Electro-static
Alignment Needs	3	0	0	-1	-1	1	1	1	0
Easy to attach	1	0	-1	-1	-1	0	0	0	0
Easy to detach	1	0	-1	-1	-1	0	0	0	0
Tension	1	0	0	0	0	0	0	0	-1
Torque	2	0	0	1	0	0	1	1	0
Sheer	2	0	0	1	1	0	1	1	1
Weight	1	0	0	0	0	-1	-1	0	0
Size	1	0	0	-1	0	-1	-1	0	-1
Ease of Manufacture	1	0	1	1	0	0	1	1	1
Cost	1	0	1	1	1	0	1	1	1
Total		0	0	0	-2	1	7	9	2

**Figure 3.11:** Pugh Chart for Comparison of Attachment Mechanisms, using electro-permanent magnets as a baseline. Both novel gripper designs perform well, with the corkscrew design standing out as the ultimate choice.

### 3.4.3 DESIGN CONCEPT SELECTION

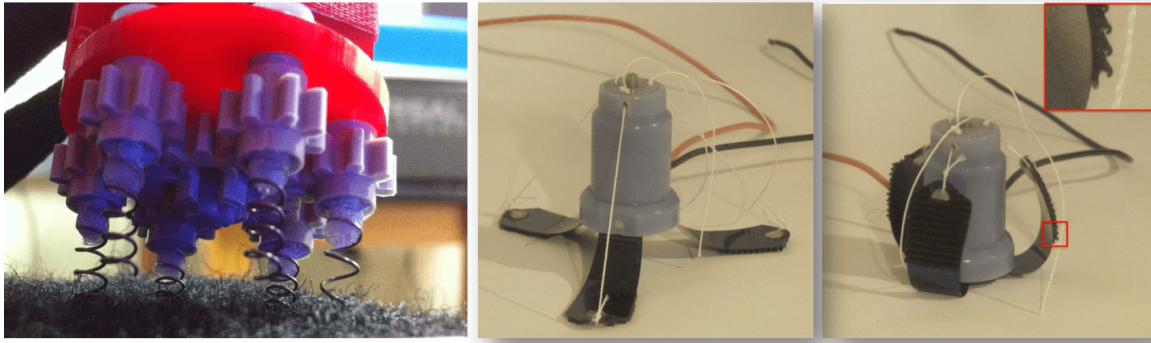
Based on the previous work, there were many possible gripper designs. Fig. 3.11 shows a Pugh chart with a variety of gripper designs as well as my ultimate choice of the corkscrew grippers.

Other gripper mechanisms and their pros and cons are discussed in detail below:

- **Electro-permanent magnets:** Electro-permanent magnets (EPMs) have a strong force to size ratio. Attachment and detachment simply requires turning the magnets on and off, and the system can remain attached without constant energy input. However, magnetic adhesion while strong in tension, is weaker in sheer and peel. While EPMs have an impressive force to weight ratio, they are strongest when connected to another magnet or EPM, and integrate magnetizing within the design would require set latching points. Creating a ferromagnetic body through molding or 3D printing is possible but the magnetic force would be considerably reduced.
- **Suction:** Active suction allows for adhesion anywhere on the body, but requires the use of a vacuum pump, which can be fairly large and heavy and would require constant energy consumption during attachment. Passive suction can be achieved with cheap, off-the-shelf suction cups, but works best on flat, smooth surfaces, i.e. not ideal for climbing on other robots.
- **Pincher Grippers:** Ants use their mandibles to pinch and grab hold of other ants, thus a pincher gripper seems like a logical, bio-inspired solution. However, most ready-to-use robotic grippers are made for manipulating small, lightweight objects, not holding an entire robot. Their size and weight are generally large compared to the gripping force, and larger gripping forces generally require either a heavier, more powerful actuator or a larger gripper for mechanical advantage. In addition, many are designed to actively grab, although a spring could be used to remain attached without needing to continually use power. Finally, pinchers have potential for sliding and rotating as their contact points are often small, though this can also be mediated with the introduction of soft materials.

- **Electroadhesion** Electro-adhesion provides a simple, low power option to attach to a variety of surfaces<sup>57</sup>. It is lightweight, does well in shear. One main drawback is weakness in peel. In addition, while robots that use electroadhesion excel on flat terrains and deal well even with surface asperities, they generally require large contact areas (e.g. for<sup>57</sup> 10 cm<sup>2</sup> is required for moving on damp concrete, the worst of the surfaces that they tested), and have not been used for curved or truly bumpy surfaces.
- **Adhesive Pads, Hooks and Micro-spines:** Directional adhesives are very effective in shear. However, they are not as good in peel and usually require a complex motion to detach (often taking advantage of the weakness in peel. Ants also use adhesive pads and tarsal hooks to grab onto one another (see Fig. 3.9. Like cockroaches and other insects which have served inspiration for micro-spines in some climbing robots<sup>71,62</sup> ants have small hairs and spines on their tarsi, which can help in gripping.

While previous designs had various advantages, none were ideal for amorphous climbing with a flipping robot. Therefore I chose to investigate two new gripper designs based on the idea of hooks, and micro-spines and Velcro-adhesion. Both gripper designs are novel design concepts with respect to previous work and are shown in in Fig. 3.12. On the left, the **corkscrew gripper** uses springs as miniature corkscrews to twist into and grab the surface material. Detachment is achieved by simply reversing the process and unwinding out of the Velcro. The second **Velcro hook grasper**, uses one-directional Velcro hooks (shown in the top right cutout in Fig. 3.12) on springy-legs, allowing for strong easy detachment as the hooks pull into Velcro loops, but easy detachment by peeling up each leg. The surface material for both grippers is Velcro loops; the corkscrew gripper was also tested with other materials including foam. For



**Figure 3.12:** Early Gripper designs. Left: An early version of the corkscrew gripper. Right: A Velcro gripper which uses one-directional hooks (shown in the top right zoomed in picture) on springy-legs, allowing for strong attachment, but easy detachment by peeling up each leg.

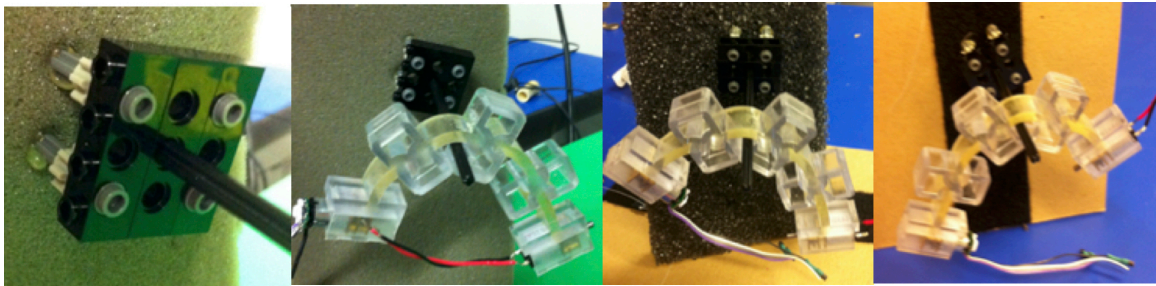
these grippers to work in self-assembly, each robot would need to be completely or partially covered in the surface material (i.e. Velcro).

Both grippers essentially rely on the strength of the loop material of the Velcro. The corkscrew gripper simply turns the miniature corkscrews to catch the loops. The corkscrews are made of small springs; as they screw in, they catch more of the material. The strength of the gripper depends on the amount of material caught by the screws. Detaching is achieved by simply running the corkscrews in reverse. The same principle applies when the corkscrew gripper uses loose foam as the attachment surface; however, this material had a much lower yield strength compared to the Velcro, and was sometimes damaged during use.

The directional Velcro gripper takes advantage of the hook material's strength in shear while using a peeling motion to detach. The hooks used are Velcro HTH719U, which are unidirectional. Thus the shear in one direction is high (35 psi according to the specifications), while in the other it is theoretically zero. The gripper design takes advantage of this by arranging the hooks to face inward; they catch when the gripper is pulled on by the body. To detach, the hooks are peeled up in the opposite direction against the curved size of the hooks. The forces required to

detach are thus very low as the actuator needs only to overcome friction and the spring force of the legs. The gripper uses a cable design, similar to that of the body of the robot. Each cable is connected to the end of a leg on the bottom, and a spool at the top, which spins and shortens the cable. Spring steel is used for the legs, and inserted into vertical slots in the 3D printed motor case in order to bias the steel to push back on the surface material, keeping the hooks in contact with the loop material, even as the gripper is pulled away from the surface.

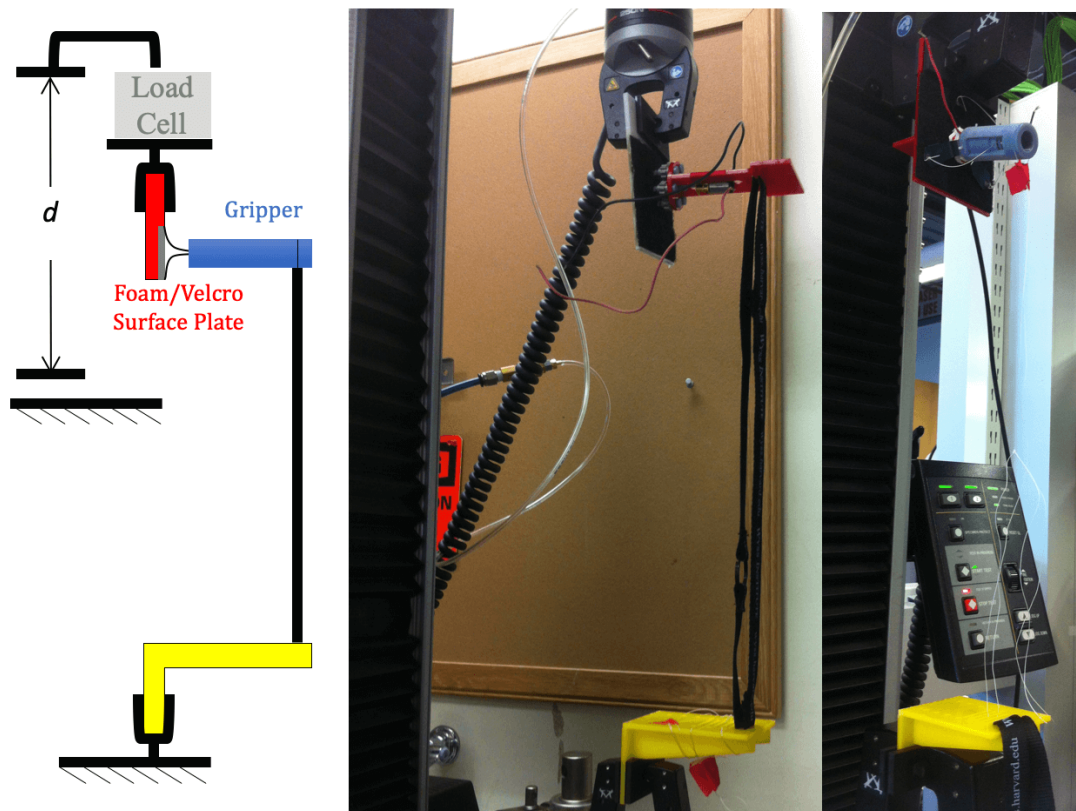
#### 3.4.4 TESTING OF GRIPPER DESIGNS



**Figure 3.13:** Testing of very early corkscrew grippers on a variety of materials - three types of foam and Velcro.

Initial testing of grippers was done by simply hanging weights from the gripper, including the robot body as shown in Fig. 3.13. This allowed me to get a baseline and choose materials to use with the two grippers.

For more rigorous testing to determine the maximum attachment force of the grippers, as well as the relation between loading force and displacement of the robot, the gripper designs were tested in the Instron set-up shown in Fig. 3.14. Both tension and torque tests were done; however, since the robot gripper is much more likely to fail under torque, I focus here on those tests. For the torque tests displacement measured on the Instron was converted to angular displacement using the set-up geometry, assuming no slip in the gripper (i.e. the gripper is a pin

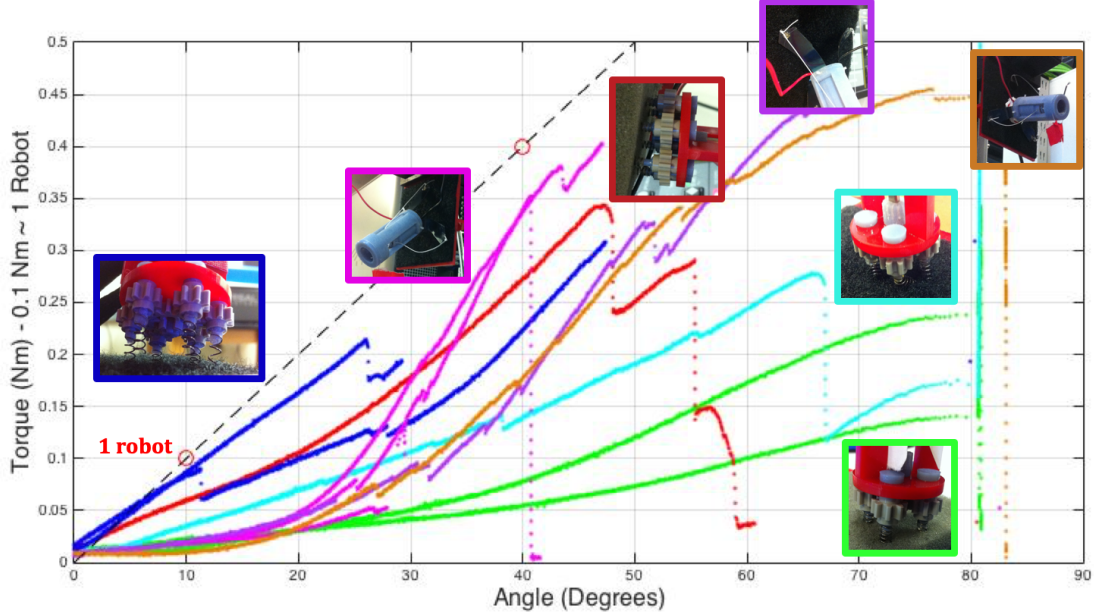


**Figure 3.14:** Setup for Instron Testing in Torque. The corkscrew gripper is shown on the left, while the Velcro gripper is shown on the right.

joint at the wall) and that all materials outside the gripper are inextensible. The ratio of angular displacement to torque is important since it represents the bending stiffness of the gripper. A low bending stiffness in the gripper will make it difficult or impossible to form cantilevers and difficult to control locomotion.

For each test, the robot gripper was placed onto an acrylic plate with the surface material (both foam and Velcro loops for the corkscrew grippers, Velcro loops only for the directional Velcro grippers), and attached via the gripper motors. The directional Velcro was placed with the legs pulled up as in Fig. 3.12 (center), and then the cable was unwound completely. The





**Figure 3.15:** An example of Instron testing results for gripper prototypes. The dashed line shows an ideal case, with the moment applied by one robot circled and noted. The closest case to this is the six corkscrew gripper on Velcro (shown in blue).

corkscrew grippers were driven with the motor to complete three turns. Once the grippers were attached, they were set up as shown, and the Instron pulled the gripper at a set speed and recorded the force from the load cell. Grippers were tested until failure i.e. until one or more springs broke or detached in the case of the corkscrew grippers, and until detachment or slippage in the case of the directional Velcro. The accuracy of the set-up was confirmed by hanging known weights from attached grippers and comparing the known weight to the recorded force from the Instron. A plot of the torque tests for various gripper configurations is shown in Fig. 3.15.

While the failure point of the Velcro gripper was high, the corkscrew gripper outperforms it when considering the ratio of torque to bending angle. In addition, the corkscrew gripper pro-

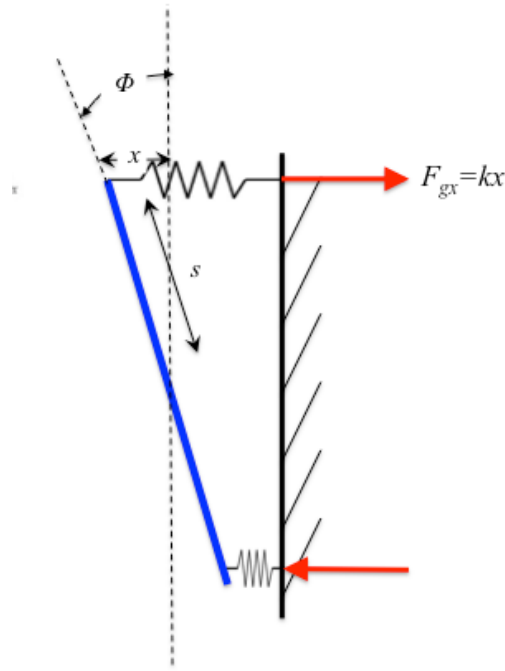
vided more opportunity for optimization. Failures in the Instron testing, as shown in Fig. 3.16 often occurred due to screws separating from the gripper, indicating a need for better manufacturing methods. In these early corkscrew grippers, I attached the springs to the gears with hot glue. In later versions, I switched to epoxy which provided a much stronger connection.



**Figure 3.16:** Corkscrew grippers often failed when springs became detached, i.e. a failure in manufacturing rather than from a failure in the underlying mechanism.

In addition to manufacturing, both grippers included several parameters that could be changed and optimized. For the legged Velcro gripper, we could adjust the size of the Velcro and the number of legs. However, this presented a cost-benefit trade off: increasing the area of the Velcro hooks and increasing the number of legs would improve the gripper only at the cost of increased size and more labor intensive fabrication. For the corkscrew gripper, adjustable parameters included: number of corkscrews, corkscrew placement, and the dimensions and material of the corkscrews themselves (diameter of the wire and overall coil, pitch, number of coils). *Importantly, by optimizing parameters in the corkscrew gripper, in particular the characteristics of the individual coils, the load capacity of the grippers could be increased, with no penalty in size or weight of the gripper.*

While additional corkscrews increase the load capacity and torque to bending ratio of the gripper, they also increase the size of the gripper. The four corkscrew configuration was also advantageous due to its small footprint which could still easily find a connection point on the Flippy robot body. This configuration was also preferred for ease of use with Lego 8 teeth plastic gears, a readily available, compact, and inexpensive way to connect the corkscrews to a central motor. The corkscrew gripper could of course be adapted for higher payloads by incorporating more corkscrews, either with a larger gripper or by switching to smaller gears, at the possible cost of ease of fabrication.



**Figure 3.17:** Gripper Model. Here we imagine moment applied to the gripper due to the weight of the robot, and the reaction forces due to the corkscrew connections.  $F_{gx}$  is the reaction force,  $\phi$  is the angular displacement,  $s$  is the y component of the radial distance from the center of the gripper to each corkscrew.

### 3.4.5 GRIPPER MODELING

To determine the desired dimensions of the corkscrews (the diameter of the coil and wire, number of turns on the cut springs), I modeled the gripper corkscrews as spring contact forces, assuming no breaking in the Velcro loops once attached. The model is shown in Fig. 3.17, where  $\phi$  is the angle of deflection,  $x$  is the change in length of the spring and  $F_x$  is the spring force. Since the robot is mostly symmetric, I ignored twisting and considered the robot in 2D, thus the distance  $s$  from the center of the gripper to the corkscrew, is the vertical projection or y component only.

Given Hooke's law and using small angle approximations, the torque exerted by the gripper is:

$$\tau = \sum ks^2\phi \quad (3.7)$$

where  $k$  is the spring constant

$$k = \frac{Gd^4}{8nD^3}, \quad (3.8)$$

$G$  is the shear modulus of the material,  $d$  is the wire diameter,  $n$  is the number of coils and  $D$  is the diameter of the coil (defined as the outside diameter of the coil minus  $d$ ). **Increasing  $k$  is the easiest way to increase the torque capacity of the gripper without increasing its size. We can increase  $k$  by choosing stronger materials ( $G$ ), a larger wire diameter  $d$ , and a smaller coil diameter  $D$ .** The main constraints on these dimensions are the size of the gripping surface substrate - the wire must still be able to catch onto the Velcro loops.

The final springs are Corrosion-Resistant Compression Spring Stock from McMaster-Carr (9663K52) with an outer diameter of 0.125", inner diameter of 0.085", and wire diameter of 0.02." Calculating the spring constant yields about 13.515 N/mm for  $n = 2.5$ . However, it is

important to note that this accounts only for the force and elasticity of the corkscrew springs, and not that of the Velcro loop. Some combination of these two will give us the actual spring constant of our gripper. While we could test individual strands of Velcro loop to obtain material properties, the connection is highly dependent on the properties of each individual connection - the number and length of the loops, twist, and so on - making it difficult and not highly beneficial to model. Therefore, I stuck to modeling the corkscrews only, since we can more easily understand their behavior and use the model to optimize the gripper.

The spring constant  $k$  increases as the number of active coils decreases, but some minimum number of coils is necessary for securing the Velcro loop. I tested springs with varying  $n$  using the same test set-up as in Fig. 3.14 and confirmed that the bending stiffness of the gripper does increase with lower  $n$ . I chose  $n$  as approximately 2.5 to increase stiffness but ensure a secure attachment to the Velcro. Lower  $n$  was also found experimentally to ease attachment and detachment.

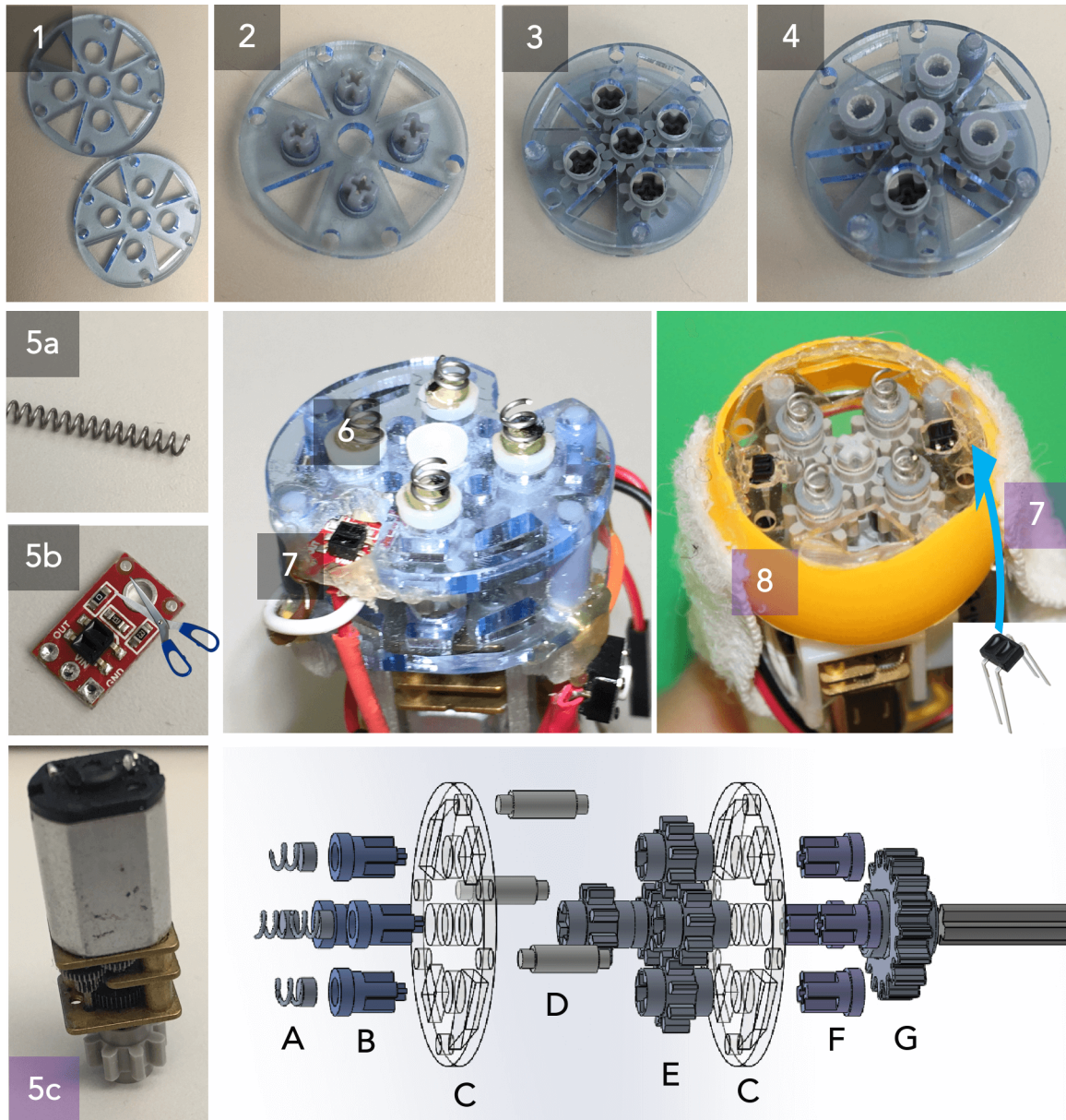
Finally, while it does not effect the load capacity, the pitch of the corkscrews is essential for attaching and detaching to the Velcro. The Velcro that we use is approximately 2-3 mm thick. The pitch of the springs was determined experimentally to be approximately 1.5 mm from edge to edge based on ease of attachment and detachment in Velcro. As no commercially available springs had such a large pitch, the springs were plastically deformed by hand until the pitch was the correct length. This process was done only once per 20" length of spring. It is possible that this deformation may cause dislocations and weaken the overall material strength of the steel. We did not notice significant differences, but if we were to manufacture these in mass, it may be advantageous to add heat-treating process to further strengthen the material of the springs.

#### 3.4.6 FINAL DESIGN AND VALIDATION

The final design of the gripper is shown in Fig. 3.7. On the following page Fig. 3.18, shows the fabrication process of the gripper and an exploded view of the individual parts in the bottom panel. The gripper consists of the corkscrews (A), 3D printed corkscrew posts (B) and (F), Lego gears (E) which connect to these posts, two acrylic plates (C) and 3D printed posts (D) for the gearbox housing, and a final Lego gear and post (G) which integrate with the motor and another gear (5C). Lego gears were chosen due for ease of use, size, and durability. The 3D printed parts are small and thus quick to print. While more complex than a simple shaft, by sandwiching the gears, the attachment force from the corkscrews is transferred along the parts to the acrylic holder and eventually the robot body. By connecting the top and bottom halves, I also ensured a degree of modularity, making the robot easy to fix if a single corkscrew broke or was deformed in the process. This was especially important during the early stages of gripper testing.

Grippers are made in about eight steps. They take a couple hours to complete and a day to cure. The glue is basic 5 minute two-part epoxy. Fabrication time can be cut significantly by scaling and completing multiple grippers in the same pass. The steps to complete a gripper are as follows:

1. Laser cut the acrylic housing (C) and 3D print the custom parts (B),(D) and (F).
2. Insert the bottom parts (F)
3. Connect the gears (E), add the posts (D) and second acrylic plate.
4. Glue the top printed parts (B) to the bottom parts (F) with epoxy.



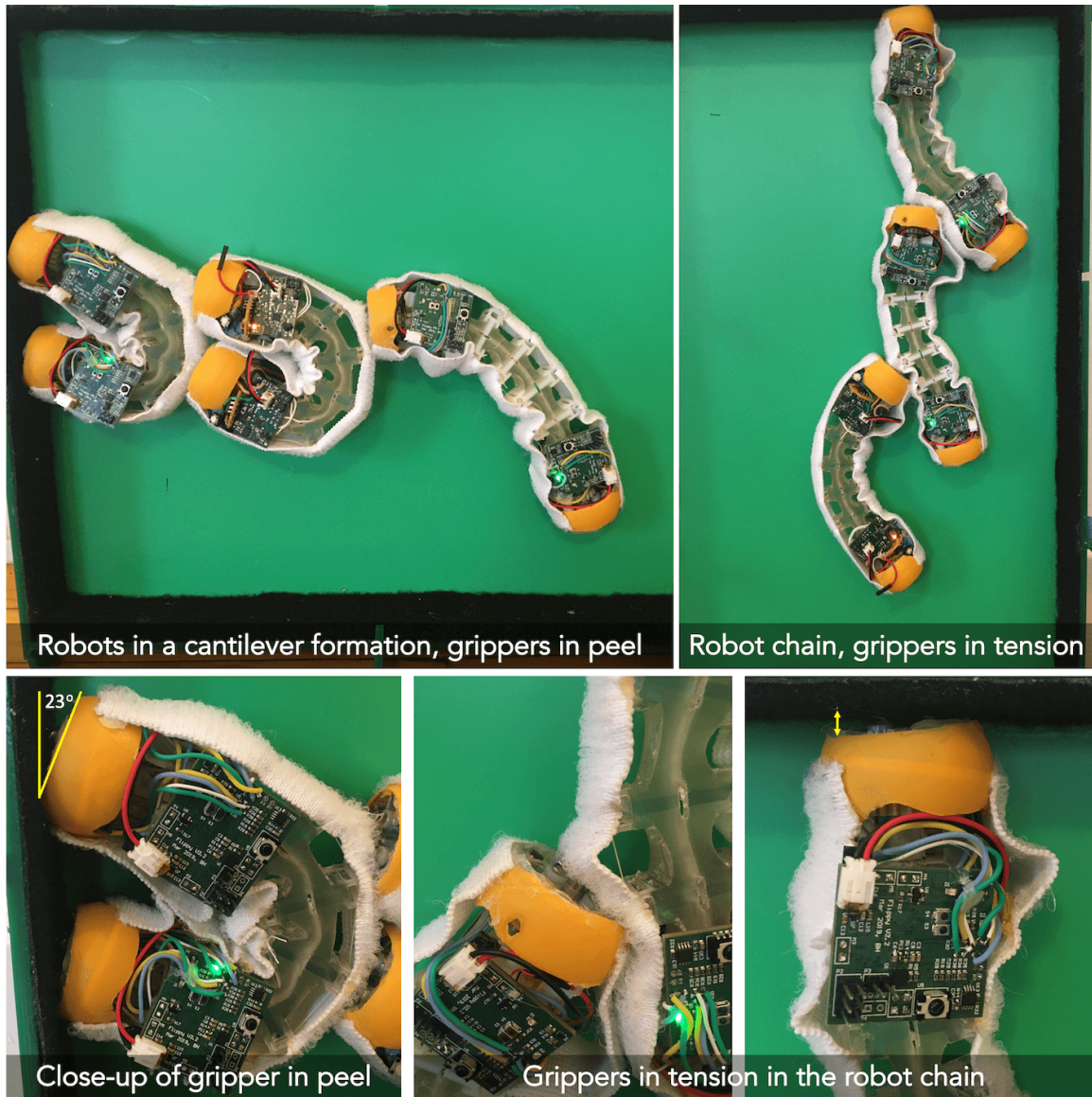
**Figure 3.18:** Gripper Fabrication and Exploded View (Bottom). Steps outlined in purple differ in the *E. robotica* robot. The Flippy connects directing to the drive train with a printed part rather than molding (5c) and does not have the orange plastic casing (8). IR sensors integration (5b and 7) also differs. 1) Laser cut the acrylic casing. 2) Add the bottom 3D printed parts (F). 3) Connect parts to the Lego gears (E) and add the gearbox posts (D). 4) Epoxy is set to glue the 3D printed top parts (B) to bottom parts (F), sandwiching the Lego gears. 5a) Corkscrews are cut to size as is the IR sensor board (b). In the *E. robotica* robot, we also mold the D-shape of the motor shaft into a Lego gear, using epoxy in the gear and mold release on the motor shaft (c). 6) Screws are molded into place using epoxy. 7) Electronics are assembled and integrated. 8) The plastic casing is cut and integrated.

5. a) Cut the corkscrew springs to size, 3.5 turns (1 full turn will be encapsulated with the printed parts). b) Cut the IR reflectance sensor breakout to size (for the Flippy robot only). c) Mold the D-shape of the motor shaft into the Lego gear, using mold release or mineral oil on the shaft, and epoxy for the mold.
6. Glue each corkscrew into place with epoxy, leaving at least 1.5 turns exposed.
7. Assemble electronics and attach sensors.
8. For the *E. robotica* robot, a plastic covering is added. The covering is made by cutting off the ends of a ping-pong ball with a regular or heated X-acto blade. Future robots could incorporate 3D printed or other coverings, but ping-pong balls provide a lightweight, robust material already molded and sized for the needs of the robot.

The final design weighs approximately 16 g, including the motor. The gripper is essentially the same for both Flippy and *Eciton robotica*, except for the addition of the plastic orange casing, fabrication improvements, motor and gear placement as discussed later in section 3.6, and small changes to sensing discussed in 3.5.2. The mechanism remains the same. While we have not tested the gripper until failure, it has held over 100x its weight in both tension and sheer (up to 2 kg in sheer and 2.3 kg in tension), without sustaining any damage. This is the equivalent of one gripper holding approximately 16 robots. While we do not actually have sixteen robots, Fig. 3.19 shows three of the final *E. robotica* robots in a chain configuration, with one gripper supporting the weight of all three robots.

Comparatively, the gripper struggles more in peel, but still achieves strong results, especially when compared to the current state of the art, where most robot grippers are not able to support long moment arms. One corkscrew gripper can consistently support the moment arm





**Figure 3.19:** Gripper Load Testing on Robots. The corkscrew grippers can allow the robot to form cantilever structures (top left) and hanging chains (top right). The bottom close-ups show that the bending angle remains small even for the large cantilever. More bending occurs in the flexible body and in the stretchy Velcro as shown in the center.

of the robot when fully extended for locomotion, as is necessary, and has been tested to support at least 1.5x this amount. If we allow both grippers to attach to the surface (as they would be in most bridge structures), we can form cantilevers as shown in Fig. 3.19 on the left, where one robot successfully manages to support the weight of two other robots, with fairly minimal bending (about  $23^\circ$  in the top gripper). It is important to note that cantilevers such as the one shown are more difficult than a typical robot structure. In most cases, additional robots could be used on the sides, to shore up the attachment point. Even the army ants themselves rarely form long cantilevers; the protruding structures we have observed in experiments were mostly limited to a little more than the length of an ant, possibly due to their flexible bodies. Ants are more likely to form droplets or chains, for which the robot gripper is well equipped.

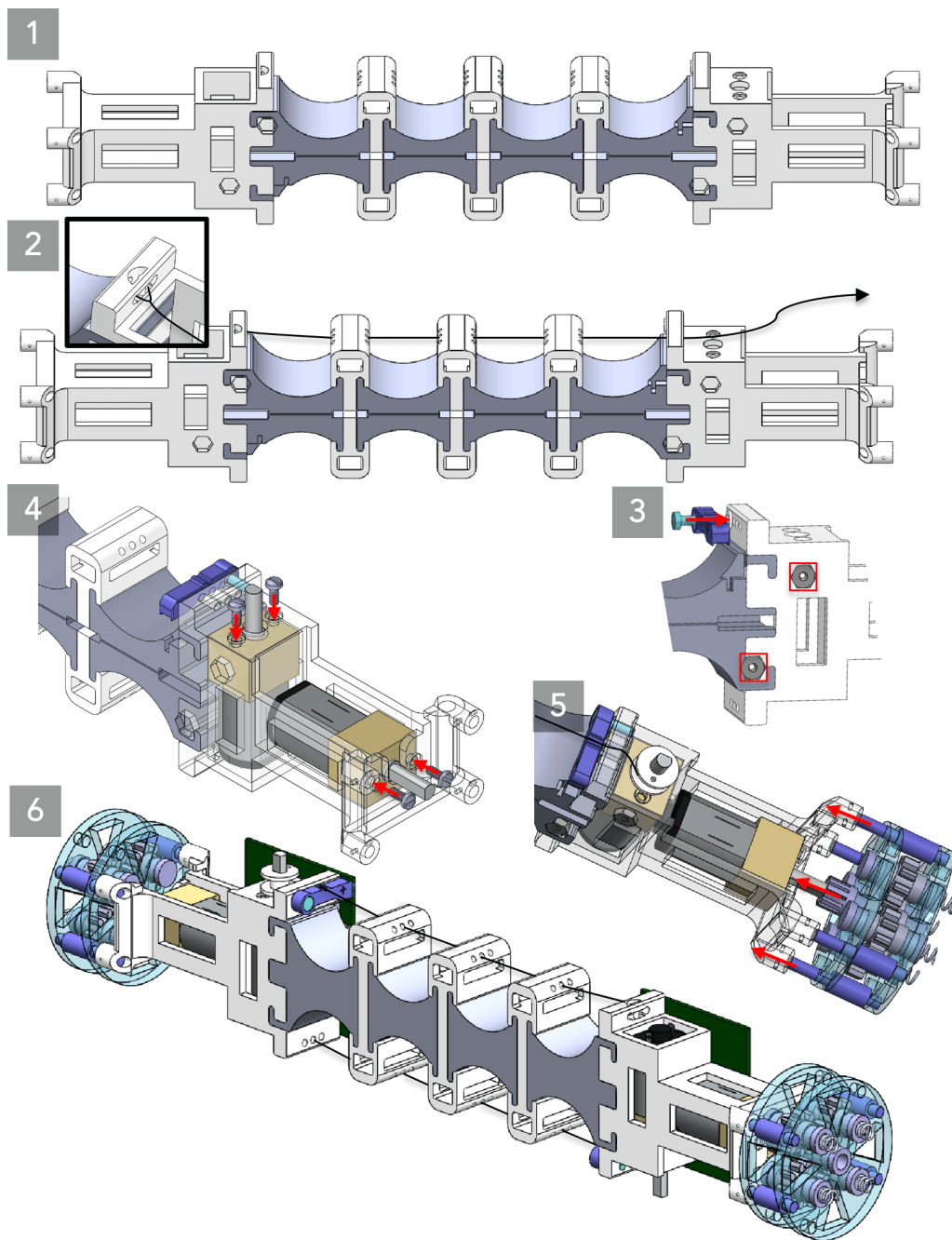
While the demonstrations in Fig. 3.19 show the potential for the gripper, the most essential test, of course, is the gripper's use on real, autonomous robots. The gripper, after all, must not only be able to withstand large loads but do so while attaching and detaching autonomously on the real robots. As described in the following sections on Flippy and *E. robotica*, as well as Chapter 5 where robots demonstrate the full self-assembly algorithm, the gripper was successfully integrated into autonomous robots and was able to support locomotion and the creation of small robot structures with connections that it made autonomously. While robots sometimes failed to connect well, and occasionally fell, this was generally due to failures in sensing rather than the gripper mechanism. The current gripper is quite robust and withstood these falls and 100s of attachment and detachment cycles without damage; *E. robotica* robots may go for months of testing, where the gripper needs very little modification or adjustment.

### 3.5 FLIPPY: A CLIMBING ROBOT

Using the body and gripper design from the previous sections, I built a soft, flexible climbing robot. To my knowledge Flippy is one of the first soft climbing robots and the only one besides Bridgebot<sup>77</sup>, which is autonomous and untethered. Due to its built-in compliance, flipping gait, and the corkscrew gripper, it could autonomously climb up and down surfaces held at any angle relative to gravity and transition from one surface to another, without complex sensing or control. The robot is composed of easy to make and off-the-shelf components making it suitable to scale for collective behavior.

#### 3.5.1 FABRICATION

The fabrication process for the Flippy robot is shown on the following page in Fig. 3.20. The body of the robot is 3D printed in VeraClear and TangoPlus from Stratasys, which allowed for quick prototyping. The pin joint and lever for the tension switch are 3D printed and press fit into a corresponding hole on the body. Size 69 Nylon thread is used as the cable, which is threaded through the stiff components and attached to a 3D printed cable spool. Gripper fabrication is shown in the previous section in Fig. 3.18. To manufacture the gripper, I plastically deformed springs to the chosen pitch of 1.5 mm. The springs were cut to length and connected glued to custom 3D printed parts via two part epoxy. These 3D printed parts connect, again via epoxy, to a bottom printed part, sandwiching the Lego gear train, and the housing which consists of two pieces of lasercut acrylic. Printed parts connects the gear train to the gripper motor (Pololu Micro Metal Gearmotor, 298:1), and the gripper to the body of the robot. The PCBs are attached to the front side of each end of the robot with the batteries on opposite side to balance the weight distribution.



**Figure 3.20:** Fabrication of a Flippy robot: 1. 3D print the body from flexible and stiff components. 2. Attach and run the cable through the stiff components. 3. Attach the tension switch on each side. 4. Assemble the motors. 5. Attach the gripper. 6. Assemble the electronics and attach the PCBs.

The final weight of the Flippy robot is 120.6 g. The body accounts for approximately one third of the weight (38 g), and the motors another third (9.4 g each). The grippers are approximately 6 g each. Batteries (4.5 g ea) and electronics account for the rest of the weight. The length of the robot is 235 mm from the tips of the corkscrews, giving the robot a total moment of 0.139 Nm (slightly larger than the original approximation, but easily within the safety factor). The length of the bending portion accounts for less than half of the length (97 mm), while the motor housing and grippers account for the rest. Later versions of the E. robotica robot reduce the length of the grippers by moving the motor to the back side instead of aligned with the robot (see Fig. 3.31).

The size and weight of the Flippy robot make it easy to handle and manufacture, while retaining portability and compactness for experiment set-up. Most of the time consuming portions of the fabrication process (e.g. printing and curing time for glued parts) do not require active human participation. Assembling the gripper and soldering the motors and other off-board components are therefore most time consuming in terms of human-hours. Adding connectors might possibly reduce this time or allow for batch production. Snap-on designs for the gripper and gear train parts could likewise decrease manufacturing time.

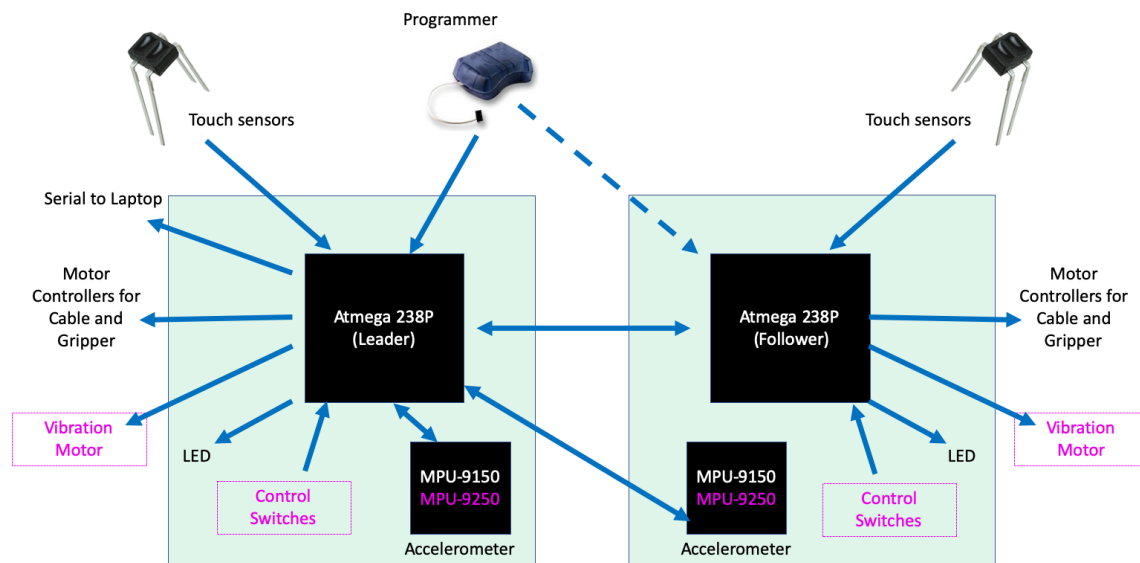
Fabrication requires only a 3D printer, laser cutter and solder iron for equipment. While I used a specialized Objet printer which can print two different materials in addition to support material, future robots could be assembled with one material printers, using either shape deposit manufacturing (SDM) techniques (printing the stiff sections and then assembling them into a mold with the flexible sections, or by printing the sections separately and assembling them. We found the TangoPlus material to be resilient and advantageous for quickly creating robot prototypes. However, the material deteriorates in the presence of UV light; to extend the

lifetime of the robot, it is essential to store them in a fully enclosed case. Stored robots were observed to last for months and even years, while exposed robots quickly developed cracks in the flexible portions of the robot.

### 3.5.2 SENSORS AND ELECTRONICS

The robot is controlled from two custom printed circuit boards in a leader-follower configuration, each with an ATmega328P microcontroller. The boards are identical except for an on/standby switch on the leader board. Each board controls two motor outputs: one that controls the length of the bending cable and one that controls the corkscrew gripper. Both are also equipped with an RGB LED and serial output for debugging. Inputs to each board include a three axes accelerometer, a magnetometer, and the mechanical switch used to sense cable tension. An IR reflectance sensor acts as a binary touch sensor for the gripper, connecting to a comparator circuit and then into a digital input on the board. The run time for the Flippy robot while flipping is approximately 50 minutes to an hour, using two 3.7 V lithium ion 150 mAh batteries (one per circuit board).

Fig. 3.21 shows the inputs and outputs for both the Flippy robot and the later E. robotica robot. Additions and changes for the E. robotica robot are highlighted in magenta. The electronics design is fairly simple and does not include new technology. It is included here for completeness and for future researchers that may wish to use similar strategies. While most inputs and outputs for the robot are fairly straightforward, the details on critical inputs such as touch and bend sensing are included below.

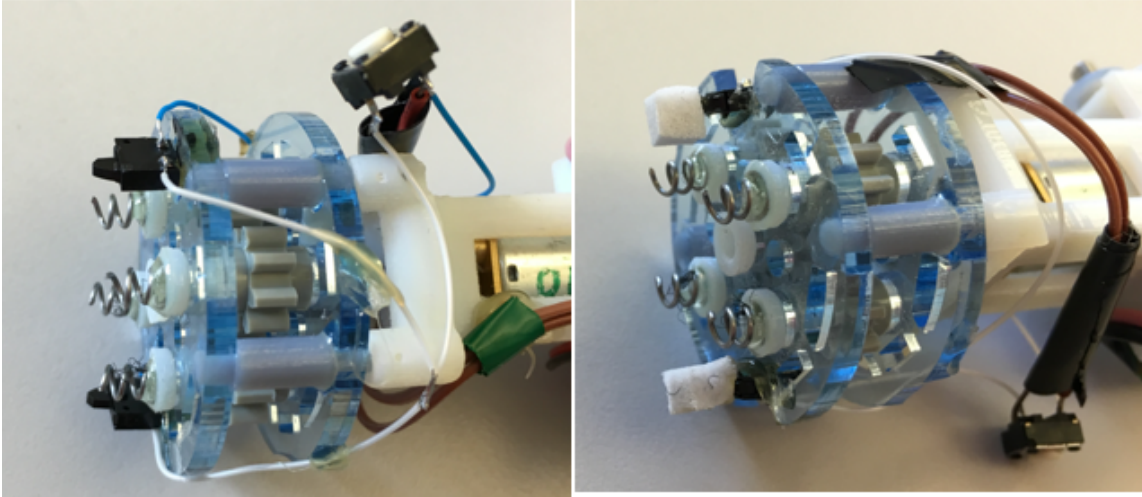


**Figure 3.21:** Block Diagram of the Inputs, Outputs and communication between the two PCBs that control the robot. Shaded portions indicate that these inputs or outputs are contained within the PCB. Components highlighted in magenta were added or updated for the final version of the robot in *Eciton robotica*.

## TOUCH SENSORS

Perhaps the most essential sensor input for the robot is the touch sensors on each gripper end which allow it to sense when it has reached a surface to attach to. The first proof-of-concept versions of the robot were controlled manually. Later, I experimented with designing a mechanical bump sensor as shown in 3.22. While I was able to demonstrate simple sensing of surfaces using this set up, it was difficult to manufacture integrate into the existing gripper. Sensitivity and tuning were difficult since the switch needed to protrude at least as far as the corkscrews, but be compliant enough to allow them to screw into the Velcro, a difficult task for off-the-shelf switches and push buttons which usually are optimized instead for smaller footprints and human touch. Moving on from mechanical sensors, I looked into electrical solutions, and chose to integrate an off-the shelf IR reflectance sensor (Pololu QTR-1A). This





**Figure 3.22:** Early Mechanical Touch Sensors, using switches.

was combined with a comparator circuit which allowed the robot to be calibrated simply with a variable resistor. The output from the comparator then went into a digital input on the board.

As discussed in the following sections, this worked fairly well for the Flippy robot. To improve reliability of both attachment and detachment, however, I added sensing capacity; for *E. robotica*, the voltage output from the reflectance sensor goes directly to an analog input on micro-controller. This allows us to use different threshold values for different states such as attachment and detachment. In addition, I decided to add a second IR sensor, which could be used to sense tilting of the gripper and develop smarter attachment control mechanisms. For better integration with the corkscrew gripper, I moved from the Pololu QTR-1A breakout to another basic reflectance sensor - QRE 1113 where I could move the voltage divider circuit to the printed circuit board. I chose resistor values experimentally, so that the sensor is most sensitive within a range of 0-15 mm, since I care only about close range contact, i.e. whether the robot can connect or are already connected.

While additional sensor feedback on the *E. robotica* robot prevented early detachment, there



were also drawbacks. Calibration by hand on the Flippy robot was simple; I simply adjusted the potentiometer on each circuit, which allowed for inherent differences in the sensors on each gripper. Calibration for the analog input proved more complex and was harder to individualize for different sensors and placement. Custom values for each sensor require more complexity in the code, and either different codes for each robot, or a table of values to draw from. As discussed further in Section 3.6 and in Chapter 5, reliable attachment remains an issue due to sub-optimal sensing.

One possible solution is for a calibration routine to be developed and run for each robot. However, auto-calibration thus far has proven to be unsuccessful. Adding a calibration routine to the set-up process for each experiment also further complicates experiment set-up, and scales poorly as robot numbers increase. I therefore chose to use set threshold values for both robots. Each robot uses a separate threshold values, but this does not account for differences between each sensor (four in total) on each robot. Variation in sensor placement may cause some of the differences in sensor read-outs. Streamlining manufacturing processes for consistent placement of sensors could also improve reliability in the touch sensors.

## BEND SENSING

As I show in the controls section and results, the robot can function as an autonomous climbing robot with *only* the binary touch sensor as input. However, it becomes much more reliable with the use of additional sensing, particularly sensing its relative orientation.

Initial versions of the robot used a flex sensor (Sparkfun SEN-10264) to sense the bending of the robot. However, these sensors are prone to wear and are sold at set lengths, so may not be easily customizable for the robot. The robot is equipped with 3-axis accelerometers on each

end; I therefore used these to sense the relative orientation of one gripper to the other. This robot bending angle, or  $\alpha$  is

$$\alpha = |\arctan(x_m, y_m) - \arctan(x_s, y_s)| \times 180/\pi \quad (3.9)$$

where  $x_m, y_m$  and  $x_s, y_s$  are the x and y acceleration readings of the master and slave sides respectively. I convert to degrees for clarity and take the absolute value for simplicity - this allows us to get the pure bending angle of the robot no matter whether it is climbing up, down or upside down. Since our system is 2D, x, and y are sufficient, but similar methods could be adapted for 3D systems.

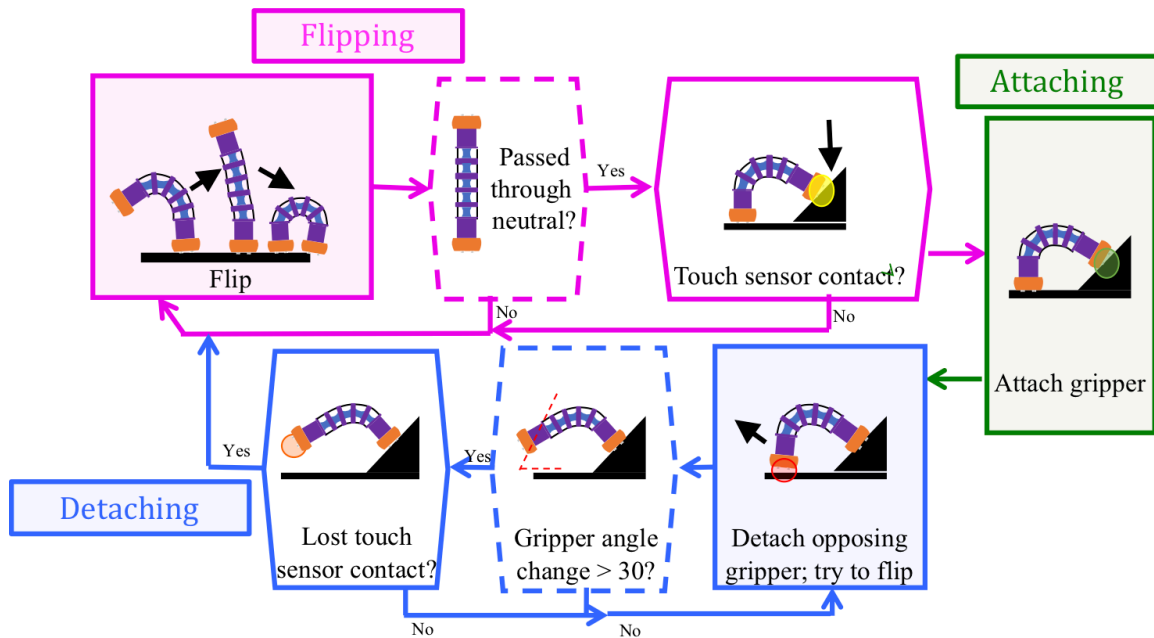
### 3.5.3 CONTROL

Many climbing robots have focused on only attachment and locomotion, assuming applications where a climbing robot can remain tethered or even be tele-operated. Untethered robots, however, will make it easier to navigate complex spaces and a swarm of self-assembling mobile robots must be both untethered and autonomous. Using only two basic sensor inputs, the IMU and binary touch sensor, the Flippy robot was able to execute autonomous flipping, climbing, and transitioning.

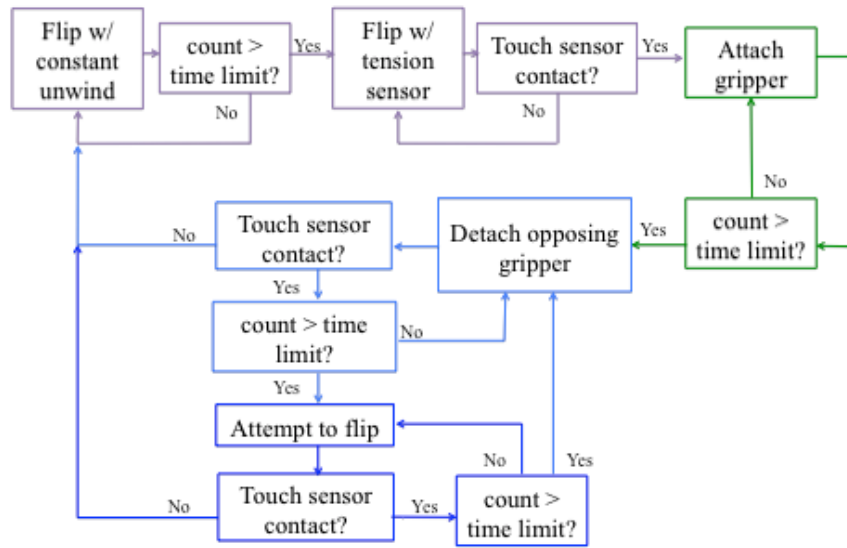
I tested the Flippy robot using two control algorithms. Both consist of a simple state machine with the following states: flipping, attaching, and detaching of each side. The RGB LED indicated the current state of the robot as well as critical checks within each states. A simplified pictorial control diagram is shown in Fig. 3.23. Dashed lines show the added checks for the second control algorithm. More details including delays are shown in Figures 3.24 and 3.25.

The simplest control algorithm uses only the binary touch sensor and is shown in Fig. 3.24.

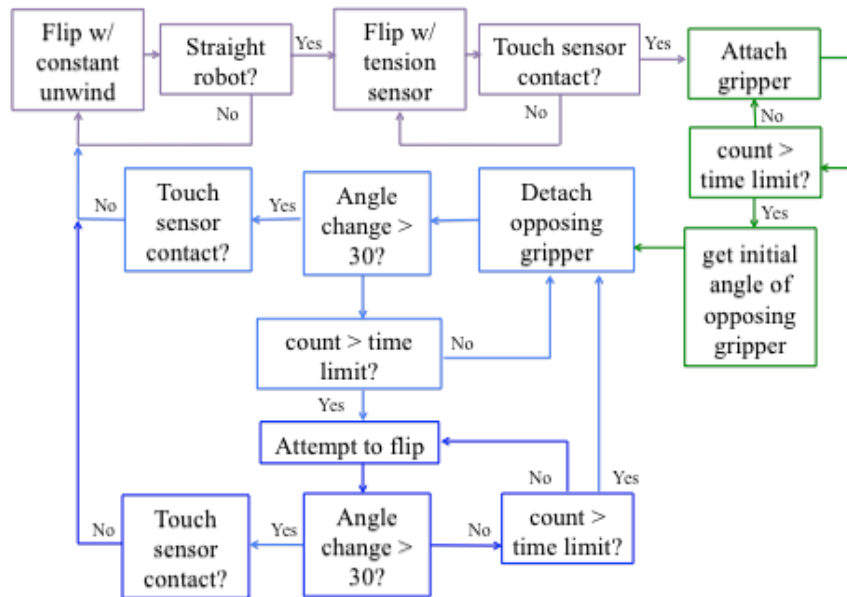
The robot starts in the flipping state, with one gripper attached and one free to move. It begins its flip by winding one cable in the forward direction, and simultaneously unwinding the opposing cable at a constant speed for a set time period. As mentioned previously, this unwinding period is useful in the case that during the detachment period the robot became twisted or bent in unexpected ways due to over-tightening of the cables. Once this period is over, the robot continues to flip, using its default method of unwinding only when it detects tension in the unwinding cable. As soon as the touch sensor detects a new surface, the robot switches to the attaching state, which merely runs the corkscrews forward (in the attaching direction) for a set period (about three seconds). It then switches to the detaching state for the opposing gripper. Here the robot runs the corkscrews backwards for a set period of time, and then tries to flip.



**Figure 3.23:** Simple Rules for Locomotion Control. States are outlined in color: The flip state is indicated in purple, attaching in green and detaching in blue. The dashed lines show the added checks using the IMU sensor for increased reliability, especially over transitions.



**Figure 3.24:** Detailed control diagram for simple control with only touch sensing including delays. The flip state is indicated in purple, attaching in green and detaching in blue.



**Figure 3.25:** Detailed control diagram for simple control with only touch sensing including delays. The flip state is indicated in purple, attaching in green and detaching in blue.

As soon as the touch sensor no longer detects the surface, the robot will return to the flipping state.

As I show in the next section, Control 1 is sufficient for basic flipping and climbing. The second control method (Control 2) shown in Fig. 3.23 and Fig. 3.25, is similar to Control 1, but uses additional sensing to achieve higher reliability when transitioning between planes. One common issue with Control 1 was that during transitions the robot would reattach to the surface before completing the entire flip. To prevent this I used the IMU as a bend sensor and implemented a check to ensure that the robot passes through the neutral, extended straight position before it attempts to attach the moving gripper. Neutral, as discussed previously, is found by calculating the orientation in the X-Y plane of each IMU and taking the absolute value of the difference between them. When this is equal to  $180^\circ$ , all sections of the robot are straight, regardless of relative orientation to gravity. Another failure case under Control 1 was caused by inadequate information from the binary IR touch sensor. Due to placement of the sensor and tilting of the gripper, especially during vertical climbing, the IR sensor would lose contact with the surface although the gripper remained attached. To avoid false positive detection of detachment, I implemented a second check: The robot records the initial orientation of the gripper it wants to detach. It then checks the current orientation of the detaching foot and will only check the touch sensor after the difference between the current and initial orientations is at least  $30^\circ$ .

### 3.5.4 CLIMBING EXPERIMENTS

#### LOCOMOTION ON A FLAT SURFACE

I first tested the robot on the flat, unconstrained Velcro loop surface, shown in Fig. 3.26a, using Control 1. The robot was placed in the center of the track at the start of each run and completed an average of six continuous flips before it drifted off the Velcro track. The first column of Table 3.1 shows the number of successful attachments, successful detachments, total successful flips, and the average time period for a full flip. The robot flipped successfully for 40/40 attempts. In 2/40 attempts, one gripper failed to detach; however, the robot could recover easily by simply detaching and reattach each gripper and trying again.

I also tested the robot climbing vertically, inverted (upside down), downwards, and the transitions between these surfaces in an open box track. The Flippy robot was able to achieve locomotion in all of these orientations; however, since the robot currently cannot steer and is liable to twisting, it was difficult to do a sustained run of more than a few flips, especially while climbing vertically.

#### CLIMBING LOCOMOTION IN A 2D TRACK

I constrained the robot to a 2D track, shown in Fig. 3.26b-h. The track is made of acrylic with a clear sliding front door and has inside dimensions of 592 x 340 x 45 mm. This allowed for 6-7 flips on the long sides and 1-4 flips (depending on orientation) on the short side, not including transitions. For some tests, the track was flipped 90°, so Flippy could do longer runs on a vertical or downwards track.

The remaining experiments shown in Table 3.1 were done inside the track. Vertical climbing



**Figure 3.26:** Locomotion and Climbing with the Flippy Robot: A) Video stills of Flippy robot on a flat, unconstrained track. (B-H) Climbing on a 2D constrained track. B) Horizontal to Vertical transition, C) Vertical climbing, D) Vertical to Inverted transition, E) Inverted climbing, F) Inverted to Downward transition, G) Downward to Horizontal transition, H) Climbing and transitioning over approximately 120° and 60° angled surfaces

**Table 3.1:** Simple Climbing Results with Control 1 (IR Sensing Only)

	Unconstrained	Horizontal	Vertical	Inverted	Down
Attachment	40	13	34	13	14
Detachment	38	13	36	13	14
Successful Flips	40/40	13/13	34/39	13/13	14/14
Avg. Period (s)	23.6	23.7	14.5	18.1	16.9

is the most difficult for the robot, thus I did additional testing in this mode for a total of 34/39 successful flips. All other locomotion modes had no failures in 13-14 flips. Both early and late detection of detachment by the IR sensor was common, especially in vertical climbing where early detection occurred in 92% of flips, due to tilting of the gripper and the placement of the IR sensor. However, these errors did not effect the robot's ability to flip, demonstrating the robustness of the design.

As shown in Table 3.1, the flip period varies somewhat depending on the angle of the surface. Somewhat surprisingly, the robot travels more slowly on flat surfaces than any other orientation and is faster going up than down. This is likely due to the time before attachment and during detachment, when the moving gripper slides along the surface, rather than the flipping period. Overall, due to the longer periods spent attaching and detaching, the robot moves slowly, at about one body-length per minute (an average of 21.5 cm/min to be exact). Since the robot moves much less than its bodylength in most flips, this is 2-3 flips per minute.

During these tests, the Flippy robot also attempted to transition between surfaces, but had limited success (0/6 attempts on horizontal to vertical, 2/7 attempts on vertical to inverted, 3/3 attempts on inverted to downward, and 3/5 attempts on downward to horizontal). While Flippy was always able to attach successfully to the new surface, it frequently had trouble de-



taching, due to the limited touch sensor data and lack of sensory input on its current bending state. Often, the robot would complete the first part of the transition to the new surface, but, on the following flip, would detect the old surface and reattach, resulting in a loop of attaching and reattaching, sometimes requiring human intervention to proceed. I added sensing capacity to the control algorithm, through the IMU, resulting in Control 2, indicated in Fig. 3.25 and Fig. 3.23 with the dashed lines. The robot must move through a certain angle to be considered “detached” and must go through the neutral position to attach.

#### CLIMBING IN A 2D TRACK WITH IMU SENSING

With Control 2, the robot was able to complete all four right angle interior transitions as shown in Fig. 3.26e-g. It was also able to complete a continuous 1.5 laps around the box track, with no errors. Table 3.2 shows the results from the transition experiments, where H is short for Horizontal, V for Vertical, I for Inverted and D for Downwards. A transition is considered successful if the robot is able to attach to the new surface, detach from the old surface, and then continue to flip along the new surface. The rate of success for all transitions improved significantly from Control 1. The robot was also able to recover from difficult transitions, for example in Fig. 3.26f, where it was unable to get a good attachment, but was still able to continue to flip downwards. Fig. 3.27 shows simple free body diagrams of the four transitions which can help us understand why some transitions are easier and some more difficult for the robot.

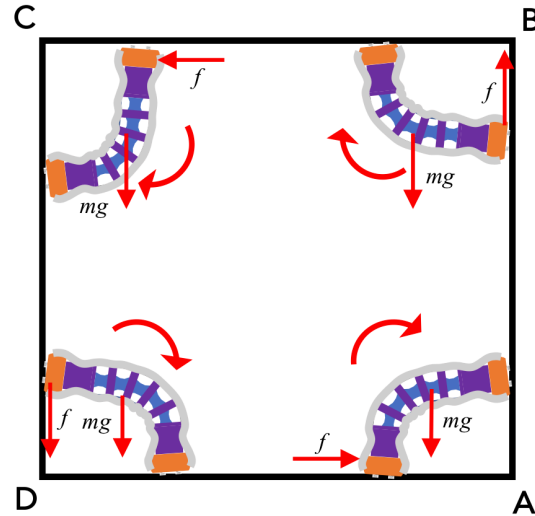
Horizontal to vertical (H to V) transitions remained the most difficult and were affected by the starting position of the robot, which was varied randomly during the trials. The most common errors occurred when the robot attempted to transition far from the vertical wall, as this resulted in lower attachment points. The lower the attachment point, the less space the robot

**Table 3.2:** Transitions with Control 2 (IMU Sensing)

	H to V	V to I	I to D	D to H
Attachment	17	13	15	16
Detachment	17	13	15	16
Continued Locomotion	9	13	15	13
Successful Transitions	9/17	13/13	15/16	13/16

will have on the subsequent flip. As shown in Fig. 3.27, the friction between the robot gripper and the surface is increased in this transition, due to gravity. In addition, if the robot gripper is low, the body will compress, adding additional forces from the elastic body to the normal force and thus the friction force on the gripper, which sometimes resulted in failure of the robot to free the gripper. These errors could be eliminated with improved sensing, as the gripper often made contact much earlier than detected by the touch sensor. In addition, in the *E. robotica* robot, I reduce friction on the gripper by adding the orange plastic casing. The change in body-length and proportions in *E. robotica*, discussed in the following section may also help, since the robot has additional compliance. Other errors included minor control problems, which were later fixed by using the completely bent position as an additional threshold.

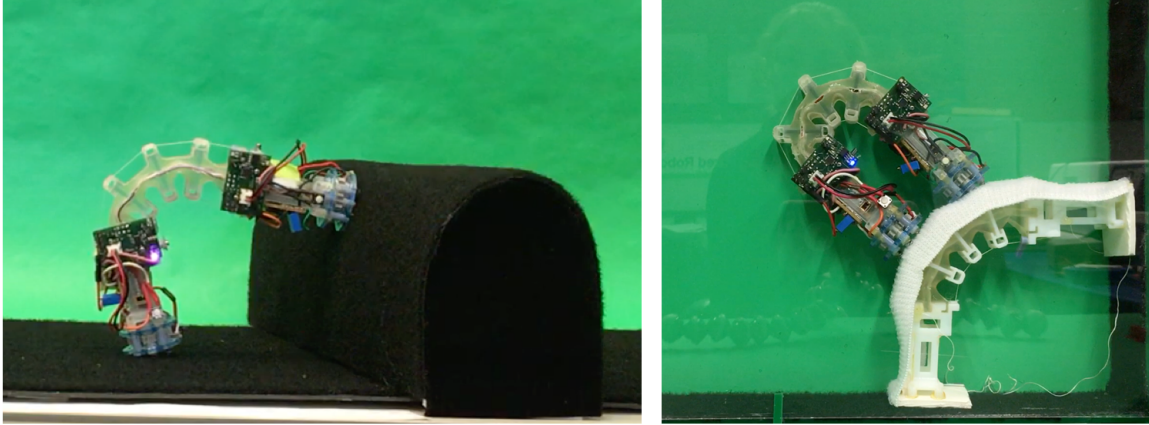
A small number of errors occurred in the other transition experiments. Three were due to attaching too close to the corner on the downwards to horizontal (D to H) transition, though this was improved after recalibration of the touch sensor. One inverted to downwards (I to D) transition also failed to attach, due to the sensors detecting the surface before the corkscrew gripper could fully reach and attach. Thus *most of the errors were due to inadequate touch sensing*. Because the touch sensors were used as a digital switch, with only one threshold for both attaching and detaching, neither attaching nor detaching could be completely accurate. In



**Figure 3.27:** Free Body Diagrams for climbing transitions. The weight of the robot helps the robot to detach in corners B (Vertical to Inverted) and C (Inverted to Down), while working against detachment in D (Down to Flat) and A (Flat to Vertical).

*Eciton robotica*, I endeavored to improve sensing on the robot gripper by adding one additional sensor to account for orientation and connecting the sensors to an analog input. This allowed for slightly more complex control through the use of different threshold values.

Finally, in Fig. 3.26h, I show that the Flippy robot can transition over surfaces held at angles other than  $90^\circ$ . This is important for many possible applications where the robot is traversing rough terrain, and has parallels in ant bridge formation which often starts where single ants can bridge a gap. Flippy successfully transitioned twice over an approximately  $120^\circ$  angle and once over a  $60^\circ$  transition, and was able to complete these transitions on a thin, flexible ramp, demonstrating that it can traverse on unstable, flexible surfaces (like a structure composed of robot bodies).



**Figure 3.28:** Initial Attempts at climbing over other robot structures by the Flippy robot.

### 3.5.5 CLIMBING OVER OTHER ROBOTS

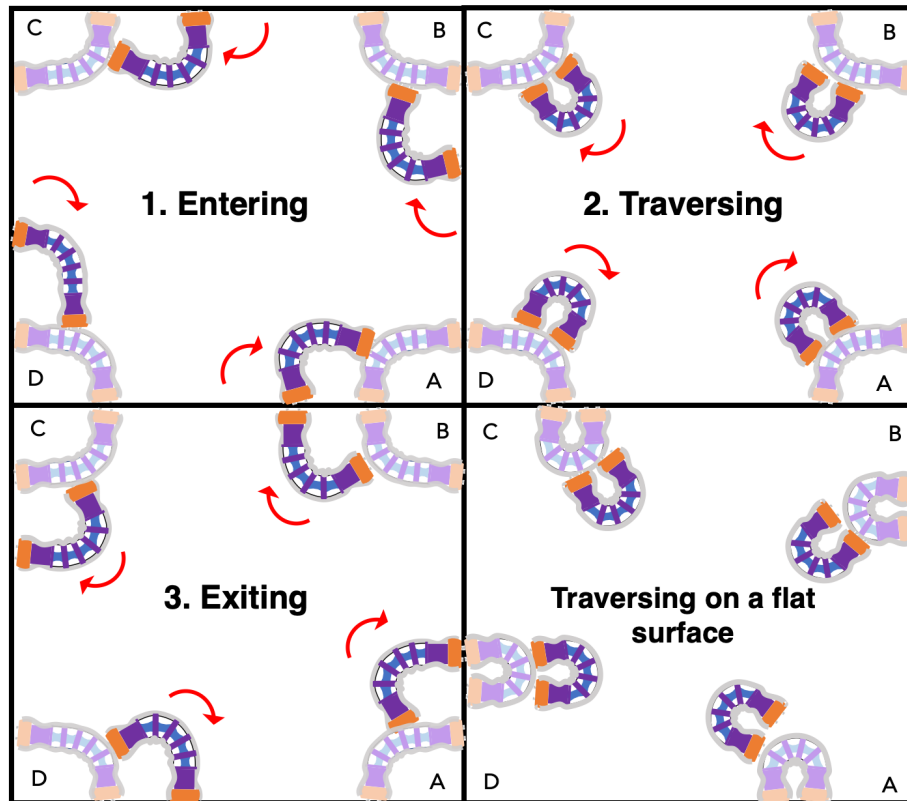
While the Flippy robot was successful in climbing consistently, it was not able to climb over itself or replicas of itself. As shown in Fig. 3.28, I first tested the robot on a fake hill, shaped to mimic the same curve as the real robot when bent. As shown, robot was sometimes able to complete part of the hill, but never completely traversed the obstacle. Since this first hill was freestanding, and the robot had a tendency to move side to side, testing was moved back to the robot box described previously. Since the robot would be climbing over other robots, I switched out the cardboard hill for a robot body covered in stretchy Velcro. I also moved to test climbing over an easier and more realistic obstacle for self assembly - where the robot being climbed over is not completely bent but transferring between planes, as shown in the left portion of Fig. 3.28. This transition position is analogous to the ant bridge building where natural bridges form between leaves or branches and in the army ant experiments in <sup>3</sup>. In the simulation described in Chapter 4, we explored robotic bridge formation in similar V-shaped terrains.

I tested climbing over other robots in this configuration in several of the four corners of the

box terrain. Fig. 3.29 on the following page shows the process traversing another robot in each of these corners, labeled A, B, C and D. The robot that would be in climbed over is faded; in the testing described here, the robot is a fake robot body, but in the implementation of the self-assembly algorithm (i.e. in Chapter 5), this is a fully functional robot that becomes part of a self-assembled bridge structure. The second robot must be able to traverse across the first robot, attaching to the structure at (1), traversing through (2) and then exiting (3). Entering the robot structure terrain (1) is the easiest step for all four corners, and consists essentially of completing a transition as in Fig. 3.27. Step 2, traversing is the most difficult, because the robot must traverse the largest angle. Here, corner D appears easier than A, B, or C because the robots motion is with gravity. Corner D is also much more likely to have a shorter step (2), with the robot sometimes skipping it completely, whereas corners A, B, and C may require an additional step while traversing.

The Flippy robot demonstrated potential, but was unable to complete climbing (i.e. steps 1-3) in any of the tests. Overall results are summarized as follows:

- The success rate while flipping was 41% in 28 flips, much lower than when climbing over flat surfaces, even vertically.
- As suspected, step 2 had the highest rate of failure, with only 2 completed successfully of 14 attempts, mostly due to lack of reach (motivating the *E. robotica* body redesign).
- Steps 1 and 3 in contrast, were successful in 67% of the attempts. Failures in these steps were exclusively due to the robot gripper catching on the stiff body sections, which was mitigated by the gripper casing and body redesign for *E. robotica*.



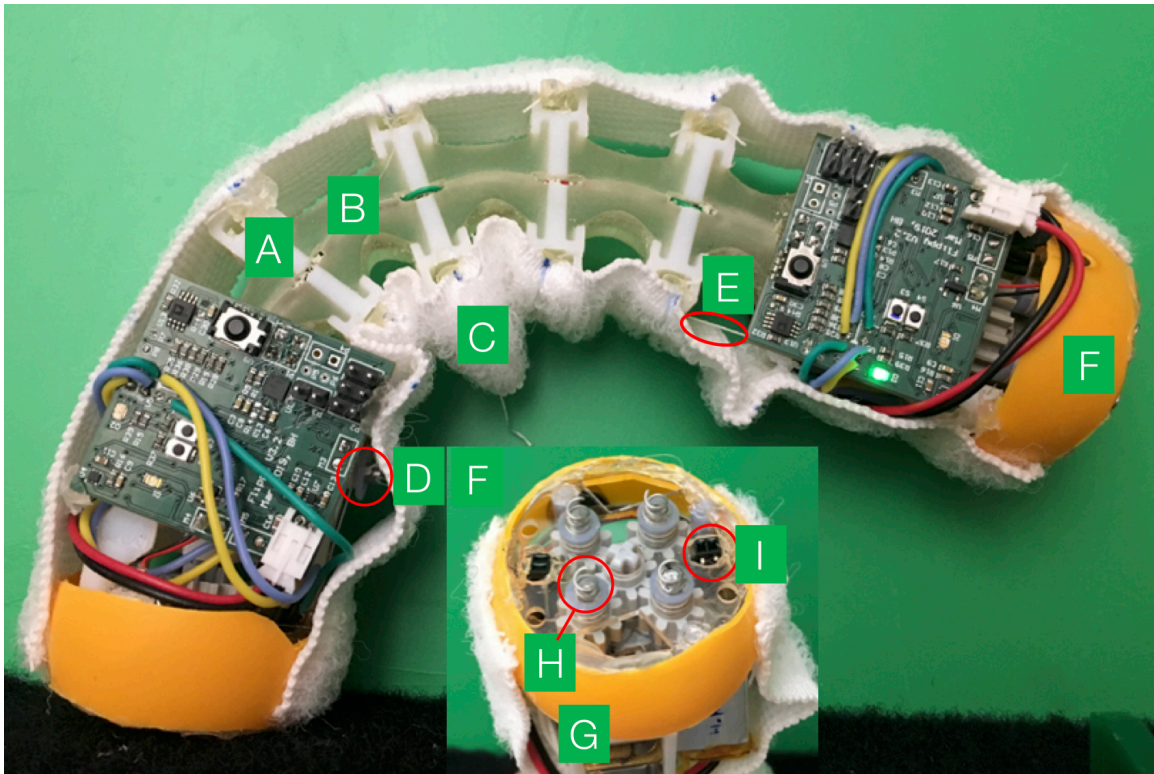
**Figure 3.29:** Climbing over other robots in corner transitions. Four corners are labeled (A,B,C,D). The robot that makes the structure and would be climbed over is faded. Three steps are required for traversing a robot in this position: 1. Attaching to the robot bridge, 2. Traversing the robot bridge, 3. Exiting. The bottom right panel shows what happens when robots traverse another robot on a flat surface; this is more difficult since the bottom robot is fully bent and it may take many flips in order to traverse (step size is smaller when traveling over the curve).

- Of the four corners, corner C was the most successful, with 75% success in flips. The Flippy robot was very close to successfully traversing the fake robot in two attempts. Failures stemmed mostly from a lack of reach (again addressed in the *E. robotica* body redesign).
- Corners B and D were the least successful, achieving no positive results. Failures in corner D were rooted in the gripper becoming caught on the stiff portions of the robot.

Since climbing over other robots is an essential part of structure building, I embarked on the re-design and adaptations discussed in the following section on *Eciton robotica*.

### 3.6 ECITON ROBOTICA

This section describes the final version of the robot modules for the hardware system *Eciton robotica*, which is able to autonomously climb over other robots. An overview of the *E. robotica* robot is shown in Fig. 3.30.



**Figure 3.30:** An *Eciton robotica* agent. The body is composed of rigid (A) and flexible (B) portions, and is covered in stretchy Velcro loop (C). A motor and spool (D) winds a cable (E) on both the top and bottom to control the bending of the robot. Grippers (F) on each end attach to velcro surfaces. A motor (G) on each gripper controls the four corkscrews (H) to attach which wind into the velcro. Two IR sensors (I) detect when a gripper has made contact with a new surface.

### 3.6.1 REDESIGN AND ADAPTATIONS

*Eciton robotica* is essentially similar to the Flippy robot with the following adaptations:

- A modified flexible body which can traverse larger curves, including angles greater than  $180^\circ$
- A reduced overall length by redesigning placement of the gripper driving motor
- Gripper covers to allow smooth travel along the bumpy surfaces of other robots.
- A stretchy Velcro covering
- Additional touch sensing capacity
- A new tension switch design
- An improved PCB

#### BODY DESIGN

The Flippy robot could traverse flat surfaces in different orientations and inner angles less than  $180^\circ$ , but was unable to climb around outer corners and convex curves. When attempting to traverse over another robot, which when bent looks like a convex curve, the Flippy robot was often unable to reach the surface. This was one of the primary sources of failure, as discussed in the previous section.

In Section 3.3.1, I described a geometric model which determined the bending of the robot based on the dimensions and number of the stiff and flexible sections and discussed the trade-offs in choosing these parameters. The maximum outer body curvature can also be determined



using some basic trigonometry. If we ignore the stiff sections which are very short and concentrate on the worst case scenario of moving from the Velcro over each flexible section, this angle in degrees is  $180 + \theta_{max}$  where  $\theta_{max}$  is the maximum bending angle of each section. This is shown earlier in Fig. 3.3 Since outer curvature is directly dependent on the bending angle, increasing the bending angle of our robot by simply increasing the maximum bending of each segment would likewise increase the curvature of the robot. In addition, as mentioned previously, the bending angle is also limited due to spacing of the grippers; they cannot overlap, which reduces the actual bending angle to less than the theoretical one provided by the model. I therefore added one additional flexible segment to the robot, for a total of five, in order to increase maximum bending while reducing the outer curvature of the robot.

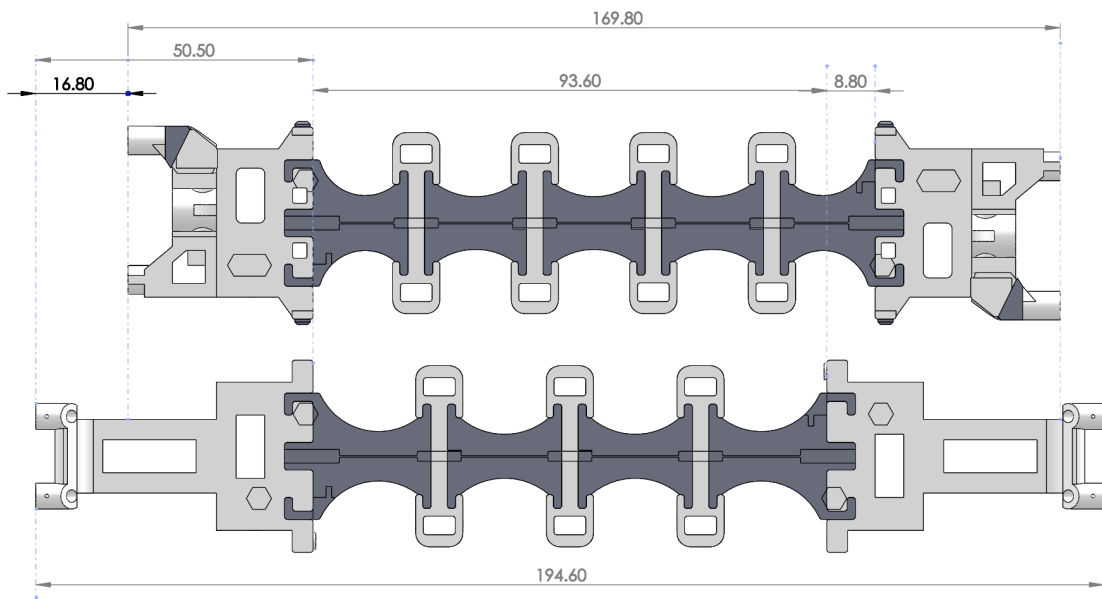
We wanted the maximum bending angle to be smaller than the maximum outer body curvature. For a five segment robot,  $5\theta_{max} > 180 + \theta_{max}$  or  $\theta_{max} > 45$ . *E. robotica*'s  $\theta_{max}$  is  $47^\circ$  for a theoretical bending angle of  $235^\circ$ . In the real robot, the measured angle is about  $230^\circ$ . The outer curvature would be  $227^\circ$  theoretically, but was measured as  $220^\circ$ , likely due to the filleted edges of

the stiff sections which allow for additional bending.

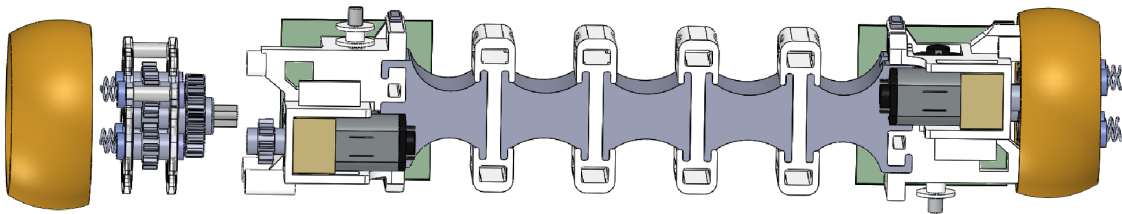
Fig. 3.31 compares the bodies of the *E. robotica* robot and the Flippy robot. Due to the additional sections, the overall length of the flexible body is increased by 9.4% in *E. robotica*. This allows the robot the larger range of motion and additional compliance to the system. Each flexible segment, however, is smaller, which decreased maximum bending angle of each segment and thus the outer curvature.

To compensate in part for the increase in length and reduce the moment arm of the robot, I reduced the length of the gripper by 16.80 mm or over 33%. This was done mostly by moving

the motor to the side and back of the end segments of the robot, which required the placement of one additional gear. Fig. 3.32 shows the backside of the robot with the gripper motor and gripper, with the motor connecting to the main gear train of the gripper from the side. Some small reductions in the flexible portion also came from the change in touch sensor design as described previously in Section 3.5.2.



**Figure 3.31:** A Comparison between the body design of E. robotica (top) and the Flippy robot (bottom).



**Figure 3.32:** Exploded back view of the E. robotica robot. The gripper motors are moved back and to the side in comparison with the Flippy robot, allowing for the overall reduction in body length.

## VELCRO ATTACHMENT

To allow the robot to both climb by itself and attach to and climb over fellow robots, the robot uses the corkscrew grippers introduced in Section 3.4 and is covered in soft, stretchy Velcro loop (C). The gripper design, shown in Fig. 3.2 (F), uses four springs as corkscrews (H) which wind into the stretchable Velcro to form a connection. A motor controls the corkscrews via a simple gearbox drivetrain in an acrylic housing. To detach, the corkscrews run backwards, unwinding out of the Velcro, and the robot alternately attempts to flip, shaking itself free. The robot can attach and detach consistently with little noticeable wear on the Velcro surface of either the ground or other robots. To help the end effectors easily slide along the bumpy surface of other robots, I enclosed the gearbox within a plastic casing and run the corkscrews backwards - in the detachment direction - while the gripper is moving.

As shown in Fig. 3.30, Velcro is attached along the entire top and bottom portions of the robot, and secured at each rigid body segment. Unlike most modular robots which attach at specified docking points, *E. robotica* can attach to its fellow agents at any point along their body (e.g. Fig. 3.34), allowing for more structural configurations. Very few self-assembling or modular robots have this capacity, which is essential for making non-lattice-based structures. Other examples include FireAnt<sup>46</sup>, another 2D climbing biped able to climb over models of itself, and Slimebot<sup>14</sup> which also uses Velcro to make formations on a flat surface.

Attachment of the Velcro proved a significant challenge, since the Velcro must provide a stable surface for other robots to traverse, while still allowing the robot itself to easily flip. As can be seen in Fig. 3.30, the Velcro stretches on the top and bunches along the bottom when bending. Both the restoring force of the elastic along the top and the bunching which may prevent the stiff portions from touching can hinder bending in the robot. The motors must

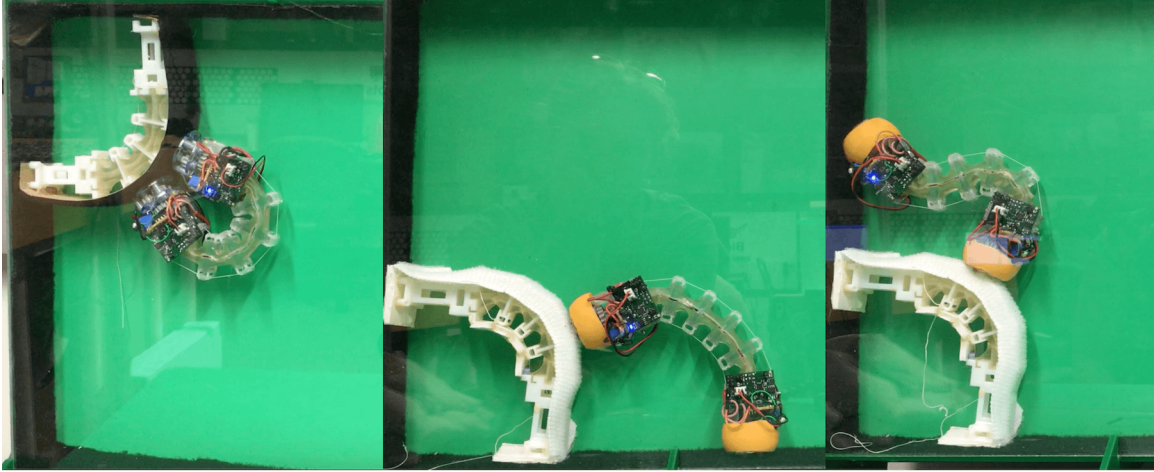
be able to overcome the moment caused by the elastic and the Velcro must be attached in a way that allows the material to move out of the way of the stiff sections as much as possible. The elongation in the Velcro due to the bending of the flexible portions ranges from 68% to an 86% percent. I performed simple tension testing on elastic on the Instron to determine the restoring force due to stretch. Importantly, the rate is linear but then becomes exponential at approximately 60%. Therefore the length the Velcro over each flexible portion of the robot needed to be respectively longer than flexible portion itself.

I tried various attachment mechanisms for the Velcro, including gluing with both epoxy and hot glue at each stiff section and sewing snaps, which were unsuccessful. In the final robot, the Velcro is attached by first measuring out and marking each section. The centerpoint of each section is attached to a stiff section using a nylon tie. To keep the tie in place, I add a small bit of hot glue. The end portions of the Velcro are glued to the orange gripper casing, also using hot glue. This process could be improved in the future by adding small hooks to the sides of the stiff portions of the robot.

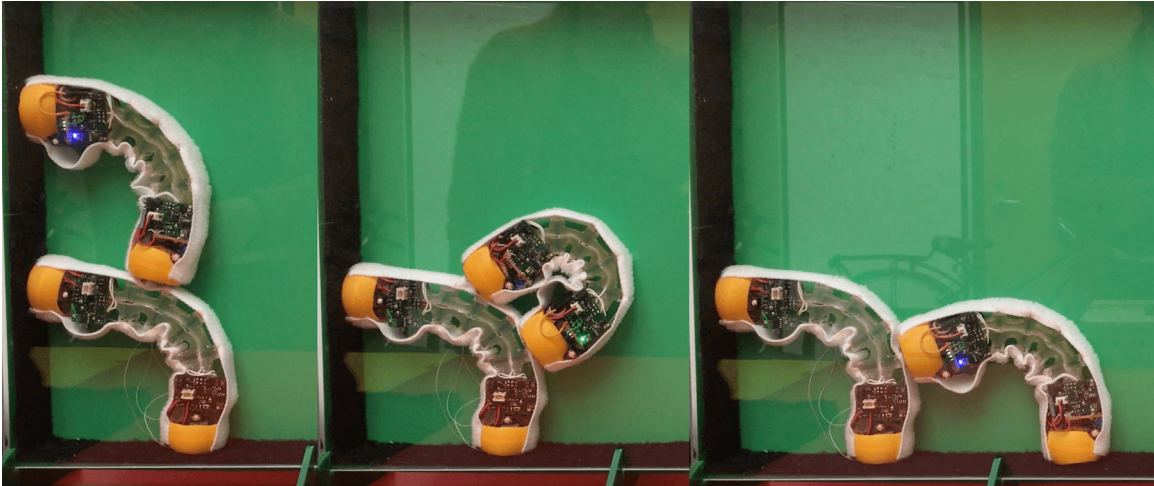
### 3.6.2 CLIMBING OVER OTHER ROBOTS - FINAL RESULTS

The final *E. robotica* robot is able to climb over other robots in several configurations. With the change in the body design only, we were able to observe a substantial improvement in climbing results.

- As shown in Fig. 3.33 on the left, **the modified body was able to successfully traverse a fake robot body in Corner C**, although this was made slightly easier by adding a stiff, continuous covering.



**Figure 3.33:** Left: The modified body design successfully climbing over a fake robot in Corner C. Center: The modified body with gripper covers successfully climbs over a fake robot in Corner D. Right: Detaching could cause S-shape deformations in the robot due to over-tensioning. This was fixed by adding another control rule so that the robot follows the ideal curve of the modeled locomotion, as shown in the figure below.



**Figure 3.34:** The *Eciton robotica* robot with Velcro covering successfully climbs over another robot, completing all three steps: entry, traversal, and exit.

- The robot achieved a substantial improvement overall, successfully completing 61% of 13 flips or a 20% increase in success rate. The success rate of Step 2 doubled, from 14% to 33%.
- In Corner D, the robot still struggled, failing on all flip attempts. The robot would get stuck on the stiff sections of the robot, so we added the orange gripper covers as shown in the center and right panels.

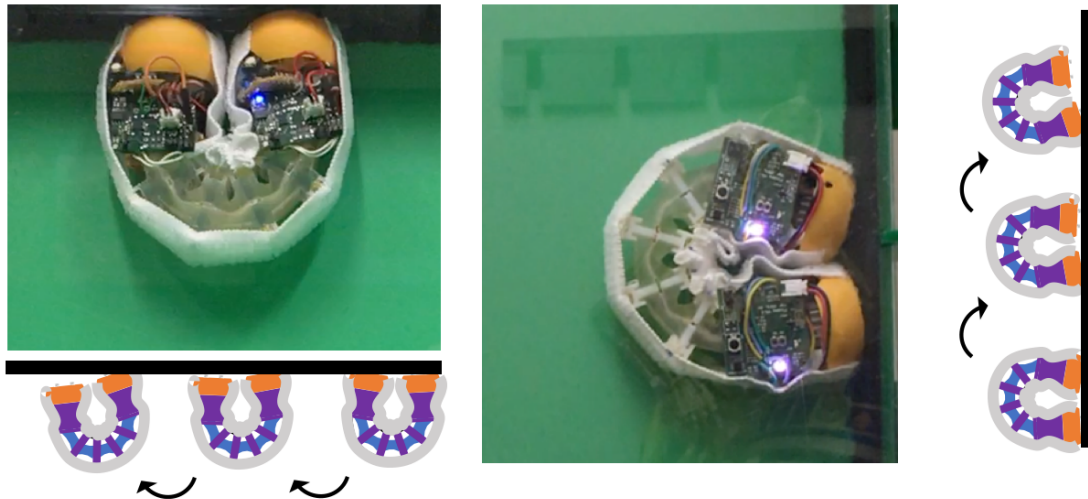
With the addition of the orange gripper covers, shown in the center and right panels, we again improved the performance of the robot.

- Overall success was 65% of 26 flips. **The robot completed the full traversal twice in Corner D, *which previously had no successful flips* and once in Corner C.**
- The success rate for all steps improved, to 88% and 83% for Steps 1 and 3 respectively and 55% (again almost doubling) for Step 2.

One remaining error in Corner D is shown in the rightmost panel of Fig. 3.33 where the robot deforms into an S-shape during detachment. This occurs in transitions from vertical to horizontal and horizontal to vertical during detachment due to the increased friction on the gripper (recall Fig. 3.27) and overtensioning of the cables. This was fixed by adding another control rule during detachment: if the robot has been trying to detach for longer than a set number of flip attempts, it will unwind its cables until it no longer feels tension in either cable. Again we use control checks to ensure that the robot follows the ideal curve of the modeled locomotion, i.e. the curve shape shown in the center panel of Fig. 3.33 or any of the panels of Fig. 3.34.

This new robot was able to successfully traverse fake robots covered in Velcro. However, it also needed to be covered in Velcro for the final implementation. As discussed above in the previous section, adding the Velcro added additional elastic forces and material barriers to bending which made flipping more difficult. In our initial attempts, locomotion success rates dropped substantially. This was especially true for climbing vertically and upside down, which had previously been reliable but require the robot to bend to its maximum range. In my first iterations of Velcro attachment, the robot could not climb at all, and struggled even at times to move on a flat surface. The final method discussed earlier in this section allows for more reliable locomotion. However, the robot still struggled more with climbing vertically and upwards, failing to flip more than three times successfully in a row, completing only 8/18 flips or less than 50% of flips in one round of testing. Travelling down or on a flat surface, it achieved success rates as high as 96% over 50 flips. We decided to focus our efforts on forming bridges on Corner D, and ensuring that the robot could complete this transition reliably. Fig. 3.34 shows the *E. robotica* robot climbing over a fellow robot in this configuration.

Since our focus was the self-assembly behavior and testing of the algorithm, it made sense to focus on the most reliable climbing configuration. However, improvements can be made to the robot in the future to allow the robot to climb more consistently and allow for structures along many terrains. The robot consistently has the same errors when in both climbing upside down and vertically, shown in Fig 3.35. Each time it flips, the gripper attachment is slightly worse and affects the subsequent flip, since poor attachment will lead to more bending in the gripper, until eventually the robot cannot reach the surface. This error accumulation also occurred in the previous Flippy robot, but was mitigated by the longer grippers, which did not allow for bending around curves, but extended the robot's reach. However, although the *E. robotica* robot is



**Figure 3.35:** Climbing Failures in *E. robotica*. Failures in climbing upside down and vertically occur as the accumulation of poor gripper attachments. As shown in the diagrams, each time the robot flips, poor attachment may allow for more tilting in the gripper on the following flip, until the robot cannot reach the surface as shown in the video stills.

only able to complete approximately 50% of flips vertically and upside down, the completed flips demonstrate that it is still capable of locomotion in these situations. As shown in Fig. 3.35, the robot is usually still quite close to the surface when it fails; in the upside-down cases, it may even sense the surface and try to attach, and then fall after an unsuccessful attempt. This indicates that if we can improve attachment and prevent attachment errors early on before they can accumulate, the robot may be able to climb these more difficult terrains more reliably.

As discussed in Section 3.5.2, calibration of the IR sensors is an ongoing issue with the robot that requires further work. In addition, as can be seen in the control diagram from the Flippy robot Fig. 3.23, and even in improved locomotion control in Fig. 5.6, the attachment state is still essentially open loop. Once the robot reaches a set threshold and is satisfied that the surface is there, it simply runs the corkscrews, and does not check to ensure that it has made a good connection. Implementing a full feedback control process for attachment - allowing the robot



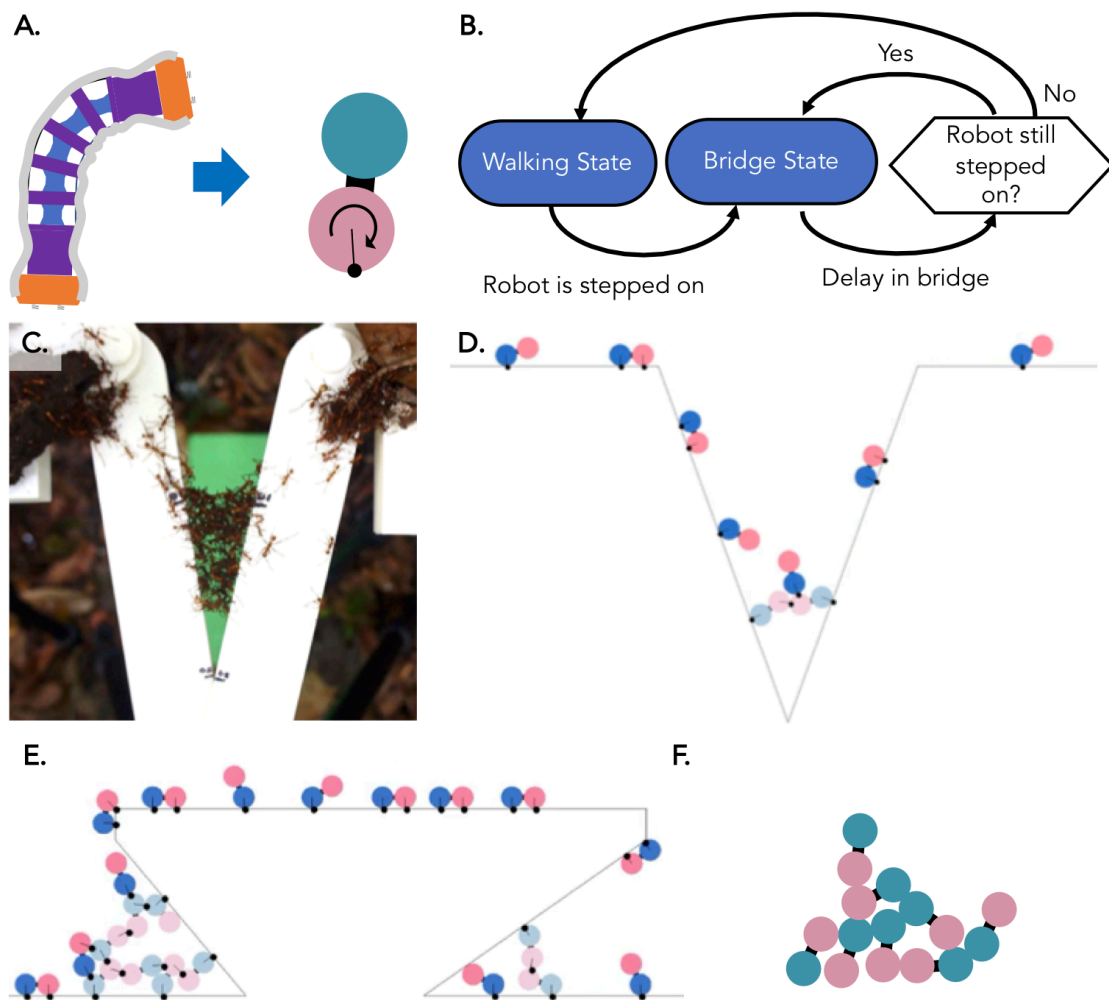
to make small flipping motions back and forth if necessary to align the gripper, and adding checks to ensure that attachment has been successful, should substantially improve the performance of the robot and allow the system to create structures on more terrains.

## Chapter 4

# Adaptive Self-Assembly in Simulation

### 4.1 OVERVIEW

In this chapter, we demonstrate in simulation that an algorithm based on local rules allows robots to self-assemble adaptive and dynamic structures. A simple abstract model of the robot (Fig. 4.1) captures essential qualities of the robot - i.e. its biped flipping gait and amorphous gripping capacity - while simplifying computation and creating a level of abstraction to demonstrate the potential of the algorithm beyond the specific *E. robotica* design. The algorithm, depicted as a state machine in Fig. 4.1B, consists of two states and relies on one main rule, i.e. robots stop moving and join the structure if stepped on.



**Figure 4.1:** A) A model composed of two rotating circular grippers captures the essential qualities of the robot hardware. B) The simple state machine for the robot algorithm. C) and D) A side by side comparison of the bridges formed by *Eciton hamatum* in<sup>3</sup> and our robot simulation. E) Simulated forming varied structures on rough terrain. F) A hypothetical simulated robot tower structure

As discussed in Chapter 2, this rule is inspired by army ant bridge building behavior. Biologists have demonstrated that army ant bridges respond to local traffic and hypothesized that ants individually experience local traffic (or lack thereof) by being stepped on or grabbed by other ants. Inspired by bridges in both nature and prior experiments<sup>3</sup>, the following simulation

work focuses on assembling bridges which allow short-cuts across V shaped terrains (C and D of Fig. 4.1), but could be expanded to other structures as demonstrated in the lower panel (ramps and towers in E and F respectively). Using a simple “join when stepped on” rule, robots in simulation create bridges that adapt to traffic: At low traffic, no bridges form; as the traffic increases, bridges form and increase in size, smoothing the path and reducing congestion. As in the natural system<sup>2,3</sup>, bridge shape and height is determined both by the traffic rate and the angle of the V-shaped terrain, and when traffic is stopped, the bridge disassembles.

In contrast to army ants, the vast majority of prior self-assembling robot systems, reviewed in Chapter 2, have focused on creating pre-determined, lattice-based shapes<sup>18</sup>. Control algorithms for these systems generally require complex path planning to move modules from one designated position to another. Some systems take a decentralized, swarm approach, e.g.<sup>6,7,87,88</sup>; yet even these rely on an outside centralized planner to determine the nature of the structure and when to form it. While this is optimal for known environments or tasks requiring precise structures, it does not allow robots to adapt in unknown or changing environments since the robots themselves do not determine the size or shape of the assembly. One approach proposed by Jing et al.<sup>89</sup> to mitigate this problem is to create a library of known assembly shapes. By using a heterogeneous collective of SMORES modules and a sensor unit, robots can sense different situations and reconfigure to one of the possible assembly shapes in response<sup>90</sup>. Kilobots similarly have made use of leader Coachbots robot to initiate the self-assembly process. These systems are an exciting step forward but lack the simplicity of swarm-like emergent behavior. They are dependent on leader or sensor robot modules, introducing a possible single point of failure, and are limited by the library of shapes, which for SMORES are lattice based due to determinate docking points.

Only a few systems in robotics have attempted insect-like functional self-assembly, where structures emerge from the reactions of individuals to their environment. In the related field of collective construction, TERMES robots<sup>9</sup> have created staircase structures, and in simulation, Melenbrink et al.<sup>91</sup> uses robots to place struts and create cantilever structures. Robots can make larger structures by reacting to local information, i.e. embedded force sensors in the struts. In self-assembly, where structures are made of the robots themselves, Swarmbots have been used to create foraging paths and form pulling chains to transport objects and to move on rough terrain<sup>13,92</sup>, demonstrating some of the first adaptive and amorphous structures. Robots stuck at the bottom of a difficult terrain or those attempting transport would signal the desire to connect via a light ring and robot structures would not consider themselves complete until all nearby robots joined. In Slimebot, inspired by slime molds, robots oscillated Velcro arms to slide and direct locomotion in the swarm, which could reconfigure around obstacles as it moves. While these systems adapted to their environment, the size of the structure was determined by the researcher - all experimental robots were part of the structure. Ants, on the other hand, adapt the size of the structure based on current needs: a large bridge will form in high traffic; at low levels, only a few ants may join. Similarly, a small gap or pothole will only require one ant, even at high traffic levels, while large gaps between branches may require a large bridge.

In our simulation, **we demonstrate the first case of dynamic, ant-like self-assembly, where both structure size and shape depend on and adapt to traffic and the environment.** Although we focus on a flipping robot and V-shaped terrain, the rules could be adapted for a wider variety of structures, such as the tower shown in Fig. 4.1f and different robot designs.

The Chapter is organized as follows, with sections 4.2-5 discussing the the simulation set-up and 4.6-7 discussing the results and conclusion:

**4.2 Simulation Selection** relates the selection of physics simulator.

**4.3 Simulated Robot Model** describes the model of the simulated robot, including its body, locomotion and control.

**4.4 Simulation Environment and Parameters** details the simulation environment and relevant parameters for building bridges.

**4.5 Algorithm** reviews the algorithm in more detail.

**4.6 Experiments and Results** discusses the results and their implications.

**4.7 Discussion and Conclusion** concludes with major lessons and expansions.

For this part of the thesis, I designed the overall simulation model and experiments and supervised several undergraduate students to create the initial simulation work in Pymunk. I later conducted the experiments in this environment discussed further in Appendix A. The final implementation and results shown in this chapter were developed by Lucie Houel, an EPFL Masters student, who I supervised. Additional details on the simulation can be found in her Master's Thesis<sup>93</sup>.

## 4.2 SIMULATION SELECTION

To simulate self-assembly using our simple rule, we needed to develop a simulation model of the flipping and grasping behavior of the robot agents, and their detection of and reaction to collisions with other agents. We needed to do this with a large number of simulated agents, with many collisions happening in parallel. For this reason, we decided to use a physics-based simulator, which can track large numbers of collisions between many rigid objects. We chose not to simulate the soft deformation of the robot body, approximating the body with rigid elements, as discussed later. This choice conserved computational power, allowing us to focus on

the interactions between robots, which are essential to the algorithm, rather than the material behavior of our particular soft design, already approximated with the simple geometric model in 3.3.1. Most physics simulators struggle to accurately represent soft materials, often approximating behavior with rigid materials and springy joints. Computing the behavior of these parts in one robot can be computationally taxing, and would substantially effect experiments with fifty or more robots.

Since the robot hardware remains in 2D, we experimented with two 2D physics simulators. While 2D behavior could be simulated in a 3D simulator such as the Open Dynamics Engine or ODE, this substantially increases computation time. However, our choice in 2D simulators was limited to two main options: Box2D and Chipmunk2. We first developed a simulation model in Pymunk, a python wrapper for Chipmunk2. Using this simulator, I conducted an initial set of experiments which showed that our rule would allow us to form bridges that responded to traffic levels and size of the terrain. Results from these initial simulations are shown and discussed in more detail in Appendix A. In Pymunk, I was not able achieve bridge dissolution except in specialized cases, where the number of robots was kept less than ten, and bridges only partially dissolved. These initial experiments helped us to better understand the underlying parameters and bridge building behavior, but the simulator had a number of disadvantages including a lack of real world units, lack of code accessibility, unpredictable behavior, and numerical errors which caused nonphysical actions. For the final implementation we moved to Box2D, which uses real world units and provides access more of the source code, allowing us to prevent possible numerical errors.

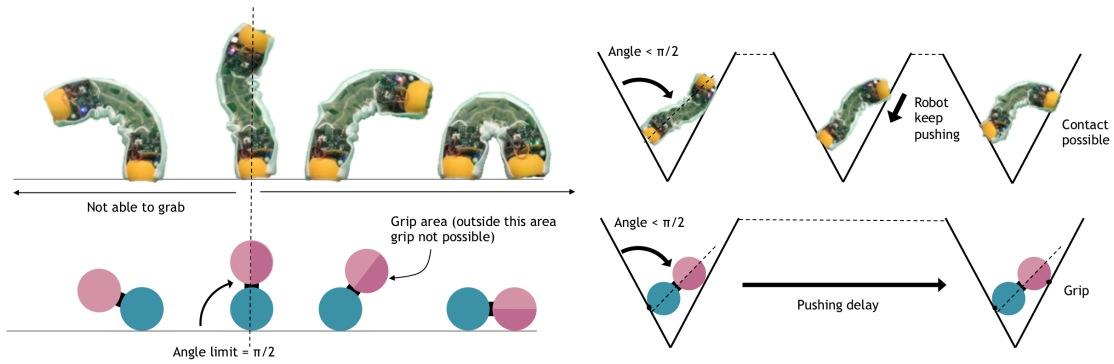
### 4.3 SIMULATED ROBOT MODEL

To simulate the effect of the self-assembly rules, robot agents needed to depict the robot body, its flipping motion, and sensing for the self-assembly state machine. We created an abstracted kinematic model of the *E. robotica* agents, which captures the main characteristics of the real robot while also allowing computationally tractable simulations of hundreds of robots.

#### ROBOT BODY

Like the system hardware, the simulated robot is a biped agent with a flipping gait (Fig. 4.2). The two grippers of the robot are modeled as rigid circles attached by a rectangular bar; we create the flipping motion by placing a motor at the joint between the circular grippers and connecting bar (at the center of each circle as shown in Fig. 4.1). The length of the rectangular connection was chosen to allow the robot to climb over a  $270^\circ$  outer angle, making it slightly more capable at traversing outer curvatures than the real robot in hardware. When the circular grippers collide with a surface, whether it is the ground or another robot, they can grab by creating a single pin joint which represents the behavior of the corkscrew gripper. Gripping is instantaneous. The gripper pin joints are compliant, which helps to approximate the compliance of real robots in hardware due to their flexibility. The pin joint may only be formed on the bottom half of the circular grippers, as shown in Fig. 4.2 as the darker gripper section. This discourages the simulated robot from grabbing robots that are above it (up-grabs), which hindered dissolution in the first Pymunk simulations, and better reflects the actual robot, where grippers are directional. Finally, additional compliance is added within the joints between the circular grippers and rectangular body, which comes into effect when robots are stationary, in the bridge state. This was useful in encouraging successful dissolution and preventing robots





**Figure 4.2:** The real robot and simulated robot flip and attach only after going through the “neutral” position, which for the simulated robot is approximated as an angle change of  $\pi/2$ . Occasionally as shown on the right, this is not possible for the simulated model due to the geometry of the environment. Thus we added a pushing delay to the simulated robot if the robot has not gone through the required angle change.

from getting stuck due to geometric constraints. We matched the compliance of this joint to that of the actual robot gripper, allowing it to move within an angle limit of  $\pm 30^\circ$ .

## LOCOMOTION CONTROL

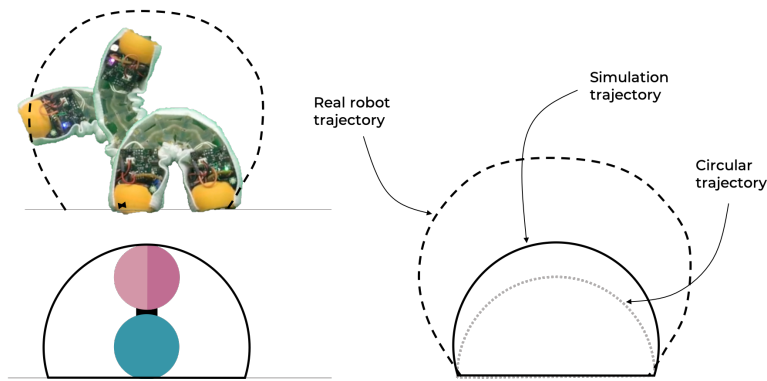
We modeled the simulated flipping locomotion and control rules after the real robot. To better model the robot, which takes time to attach and detach, and to increase the probability of being stepped on in a “good” (i.e. completely attached) position, we added a waiting delay after attaching, so that the flipping locomotion consists of a flip time and a delay time. For simplicity, the flipping time and wait time are equal.

As in the robot hardware, and shown in 3.23, when the robot starts flipping, it grabs only after passing through the “neutral,” unbent position. In hardware, this rule prevents the robot from reattaching to the same surface after detaching; the robot must always complete the first half of its flip before finding a new surface. However, unlike the robot in hardware, where the robot bends and neutral has a clear definition as a straight, unbent position, the simulated

robot is always perfectly straight. An angle limit therefore was created as an analog, requiring that the robot always pass through an angle of  $\pi/2$  before trying to grab as shown in Fig. 4.2. As shown on the right, this is not always possible for the simulated model due to the geometry of the environment, so we added a pushing delay to the simulated robot if the robot has not gone through the “neutral position.” If the robot collides with ground or another robot before reaching  $\pi/2$ , it has a random delay before grabbing, which is proportional to its angular distance from  $\pi/2$ . This matches well with the hardware system, which in tight angles may make contact at or before its neutral position (as in Fig. 4.2 on the top right), but then will continue to flip, conforming to the surface until the gripper makes contact and the robot can attach.

In addition to the neutral position pushing delay, the robot must flip and change its rotational angle by at least  $\pi/8$  radians or  $22.5^\circ$  before gripping. This minimum change again prevents the robot from reattaching to a surface in the case of slip, and is analogous to the angle check in the detaching phase of the robot in hardware as discussed in 3.5.3 and shown in Fig. 3.23.

Overall, we created a robot model that nicely represents the flipping locomotion of the real robot hardware. One difference between the two systems is the trajectories of the robots in the



**Figure 4.3:** The flipping trajectory of the simulated model of the robot differs slightly with the real robot hardware.

flipping motion, with the real robot reaching farther after straightening, and closer when fully bent, as seen above in Fig. 4.3. This may change the relative distance where robots collide but does not alter the overall assembly behavior of the robot.

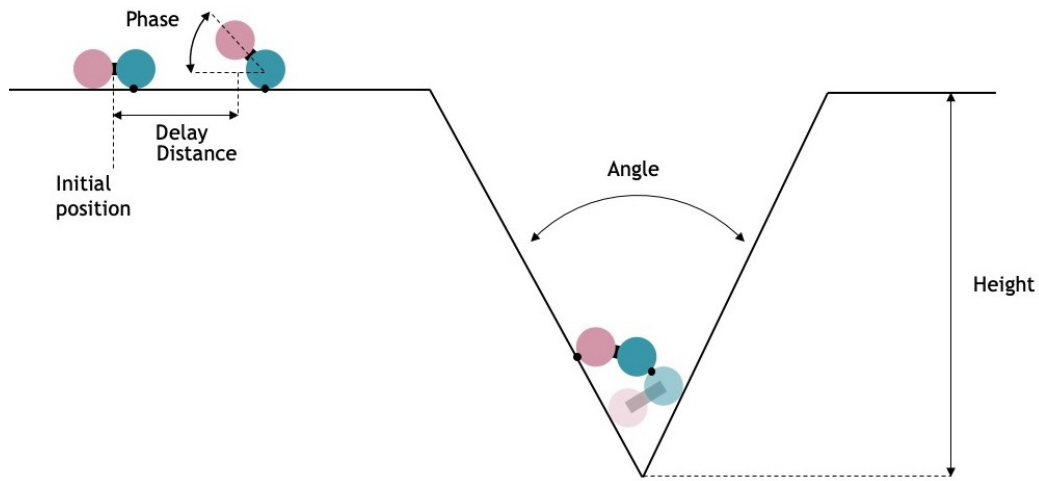
## SENSING

A simulated robot can sense if it is being grabbed by another robot, implemented in the hardware system through vibration pulses as described in Chapter 5. For simplicity, in simulation, sensing is done via the collision detector in the physics simulator. When a collision is detected, robots are designated as “contactor” or “contacted,” depending on whether they are moving (“contactors”) or stationary (“contacted”). “Contacted” robots will enter the bridge state while “contactor” robots may continue to move, except in the case where both robots are “contactors” in which case both robots enter the bridge state, and the robot that has moved more than 90 degrees is allowed to form a gripper and attach to the other robot. Additional details including the decision tree are included in Houel’s thesis<sup>93</sup>.

## 4.4 SIMULATION ENVIRONMENT AND PARAMETERS

Bridge formation and dissolution experiments were simulated on a vertical terrain with a V-shaped gap (Fig. 4.4), inspired by the experiments in<sup>3</sup> shown in the middle panel of Fig. 4.1. These ant experiments demonstrated *Eciton hamatum*’s response to differently angled terrains.

In our simulation, two sets of parameters defined the setup of each experiment: geometric parameters for the shape of the **terrain** and **traffic** parameters based on the flow of robots. For the terrain, we considered symmetric V-shaped terrains of fixed height where the angle at the

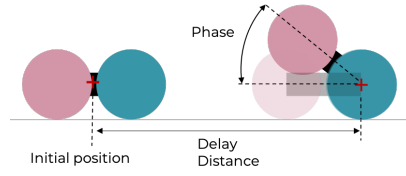


**Figure 4.4:** Simulated Set-up and parameters. Robots are spawn on the left and move from left to right.

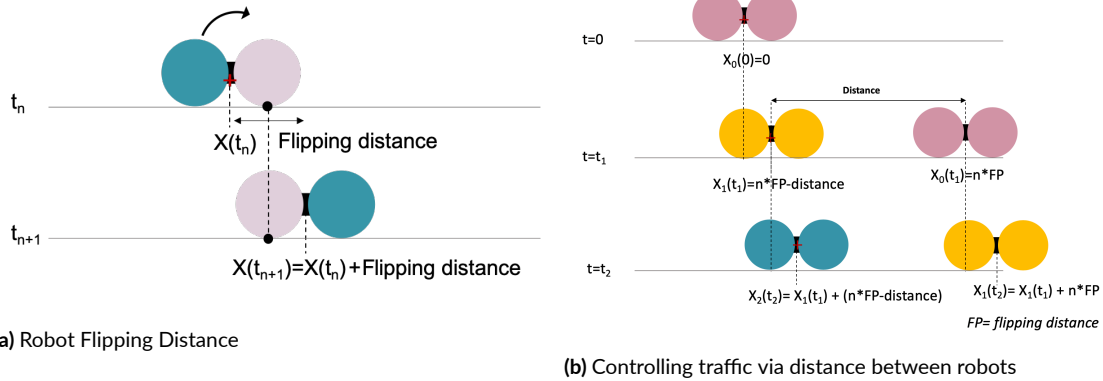
bottom is varied. Robots are created on a flat runway on the left of the terrain one after the other and move at a constant speed from left to right, exiting on a second flat terrain.

Due to the flipping gait which creates a flipping phase, the traffic flow rate is defined by the distance between the robots and their relative phase as shown in Fig. 4.4 and up close in Fig. 4.5. This can also be expressed in time domain as the time delay between robots, which will create a consistent spacing and phase difference between robots if robots are spawned at the same initial position. **Thus we can control traffic flow either by defining the robot's initial position and a time delay between spawns or the distance between robots and relative phase.**

As shown on the left in Fig. 4.6, the robot moves in discrete steps or flips. Flipping distance is generally the distance between the centers of the grippers. The trajectory of the robot is a function of the robot's initial position and the number of flips  $n$ , multiplied by the flipping



**Figure 4.5:** Traffic can be described by the distance and phase between robots and control either by these parameters or by the initial position and delay between robot spawns.



**Figure 4.6:** Left: The robot moves in discrete steps or flips. Flipping distance is generally the length from the center to center of the grippers. Right: If we directly control the spacing of the robots, the initial position and thus the path of the robot will vary, unless the distance is a multiple of the flip distance.

distance (FL). Robots created at the same initial position will travel the same path, unless there is a change in the terrain (e.g. a bridge is formed). Therefore the distance between robots when controlled via the initial position and a time delay must always be a multiple of the flipping distance plus the phase difference. This gives us fewer options when conducting a sweep of traffic parameters.

In contrast, if we control the robot traffic via a specified spawn distance, as demonstrated on the right in Fig. 4.6, we have direct control over distance and phase and thus a wider range of traffic parameters to use. However, this will change the initial position and robots will no longer follow the same trajectory. This method is also more difficult to implement. While we can place robots on a runway at set distances (as in the set-up in the robot hardware experi-

ments), once they are moving, we must calculate the spawn position and phase for the new robot based on the current robot's new position as shown in the Fig. 4.6b.

**To simplify calculations and create consistency, we chose to keep the initial position constant and control the traffic flow via the time delay.** To determine phase, the robot creation delay is drawn from a Gaussian distribution resulting in a Gaussian phase shift of  $t \mapsto \mathcal{N}(\mu, \sigma^2)$  where both  $\mu$  and  $\sigma$  are equal to  $\pi/2$ .

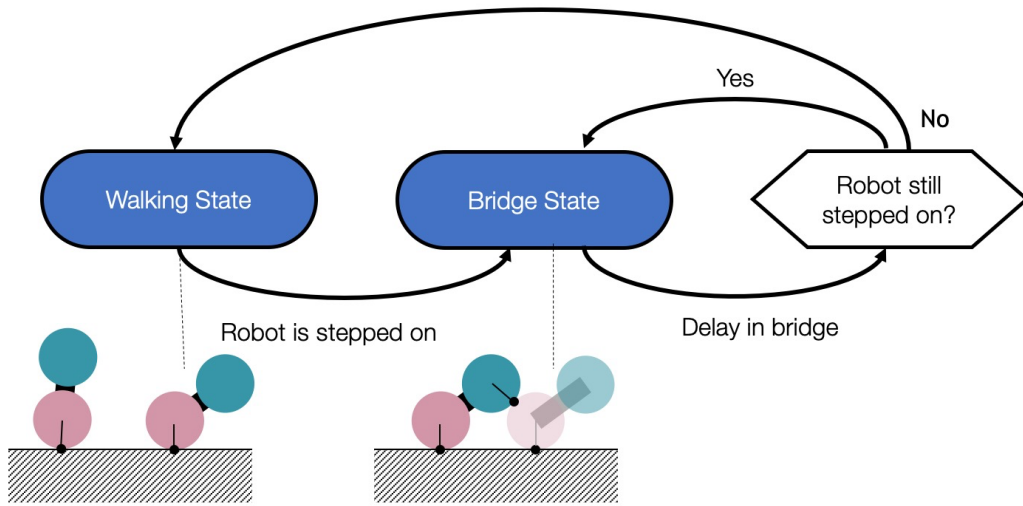
To generalize results, we use the simulated robot body-length (BL) as the length unit for both traffic flow (distance between robots) and the geometry of the terrain. We chose this over flip distance or step size, which is a function of body-length (approximately 0.5 BL). Time units are shown in seconds or flips; one robot flip period is equivalent to 1s (0.5s each for flip and wait time).

In our simulation experiments, we used these control parameters to vary traffic and terrain and study the effects of these parameters on bridge building behavior. Intuitively and from the ant experiments, in conditions of narrower V gaps and/or high traffic, we expect robots to collide more frequently and situations of congestion to occur, causing bridge formation. V gaps with smaller angles take fewer ants and also fewer robots to span; thus bridges should stabilize closer to the top of the V, smoothing the path for the other agents.

#### 4.5 ALGORITHM

The simulated robots behave according to a simple robot algorithm (Fig. 4.7). At each point in time, a robot can be in one of two states: a walking state or a bridge state. Each robot starts in the walking state and keeps flipping until it is grabbed by another robot. When grabbed, the robot transitions to the bridge state and becomes stationary. A bridge state robot only transi-

tions back to the walking state after it senses for a set time that it is no longer grabbed by any other robots. The Box2D simulator generates all of the collision events, so the dynamics of bridge formation emerge as a result of a series of collisions. We can vary one parameter of the control rule: how long the robot chooses to wait after being stepped on before trying to move again. This must be longer than robot's flip time and for the simulation, we chose it to be 5s or the equivalent of 5 robot flips.



**Figure 4.7:** The state machine for a simulated robot consists of two states: (i) the walking state where the robot continuously flips and moves forward, and (ii) the bridge state where the robot remains stationary as a result of sensing being stepped over.

## 4.6 EXPERIMENTS AND RESULTS

We studied the formation and dissolution of bridge structures across a variety of terrain shapes (i.e., V-shaped gaps of different sizes) and traffic rates (i.e., inflow rate of new robots). For each experiment, we modeled two phases (i) the formation phase, where new robots are created in the simulated environment, and we expect structures to form, and (ii) the dissolution

phase, where traffic is stopped (no new robots are created), and we expect structures to dissolve. The length of each phase (200 s) was experimentally determined to be significantly larger than needed to observe robust structure formation and dissolution.

We ran a parameter sweep varying the **terrain** and robot **traffic**.

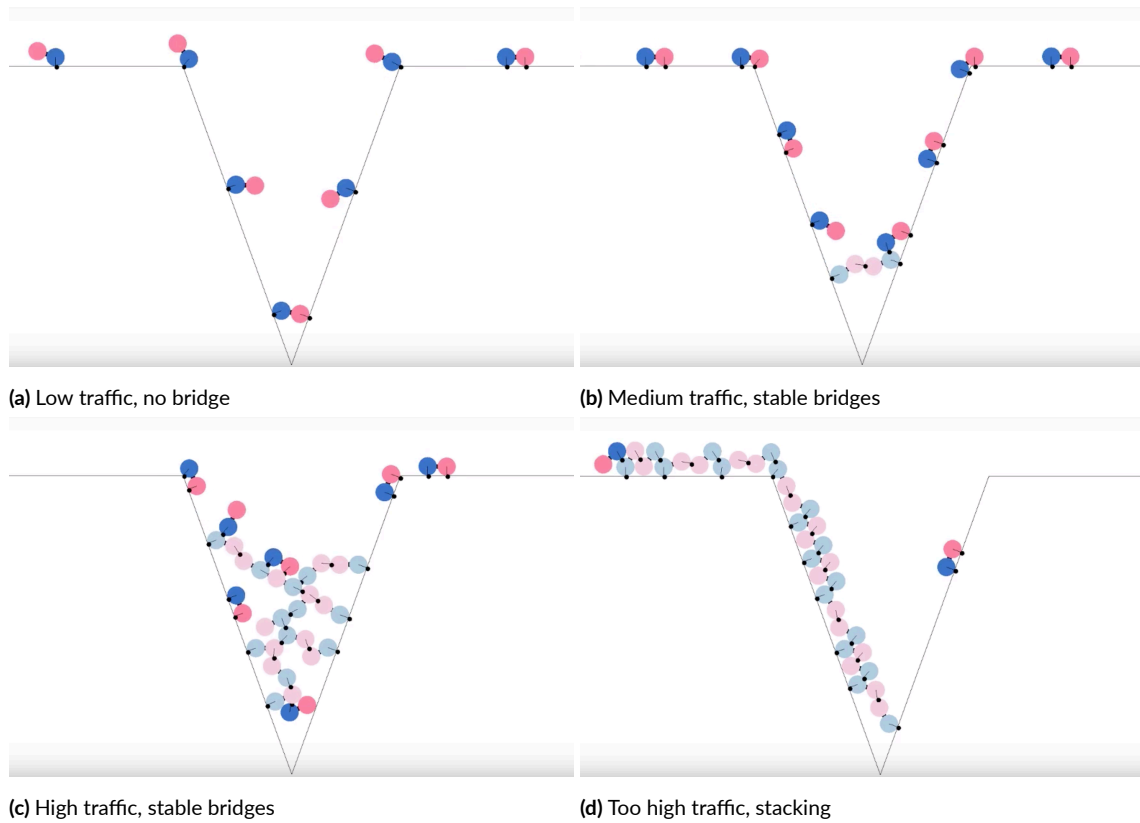
- **Terrain:** The V-shaped terrain height was fixed to 8 body-lengths (BL) and the gap angle ranged from  $20^\circ$  to  $100^\circ$  in increments of  $5^\circ$ . At  $20^\circ$ , the width of the top of the V is close to 2 BL, making collisions very likely even at the top of the gap, while at  $100^\circ$  even collisions at the bottom of the V become unlikely.
- **Traffic:** The traffic rate was defined by the delay between creation of the robots, which determines the distance in body-lengths between the robots. The traffic was varied to six different rates by setting the mean of the Gaussian distribution to values ranging from 1.3BL to 3.8BL in increments of 0.5BL. Distance is measured between centers of robots: a 1.3BL traffic mean implies that robots are practically colliding even on a flat terrain, while at greater than 4BL, traffic generates no collisions.

For each pair of terrain shape and traffic rate, a total of 10 experiments were run, summing up to a total of  $17 \times 6 \times 10 = 1020$  simulated experiments.

## BRIDGE FORMATION

Four different bridge formation types were observed across all trials. Fig. 4.8 on the following page depicts formations for a single terrain (V-angle  $40^\circ$ ) under different traffic conditions (1.3BL to 3.8BL). In the case of low traffic, no collisions occur and no bridge is formed. As the traffic increases, robots start to collide and small bridges form. The small bridges have the effect of smoothing traffic and eliminating further collisions, so these bridges remain stable for





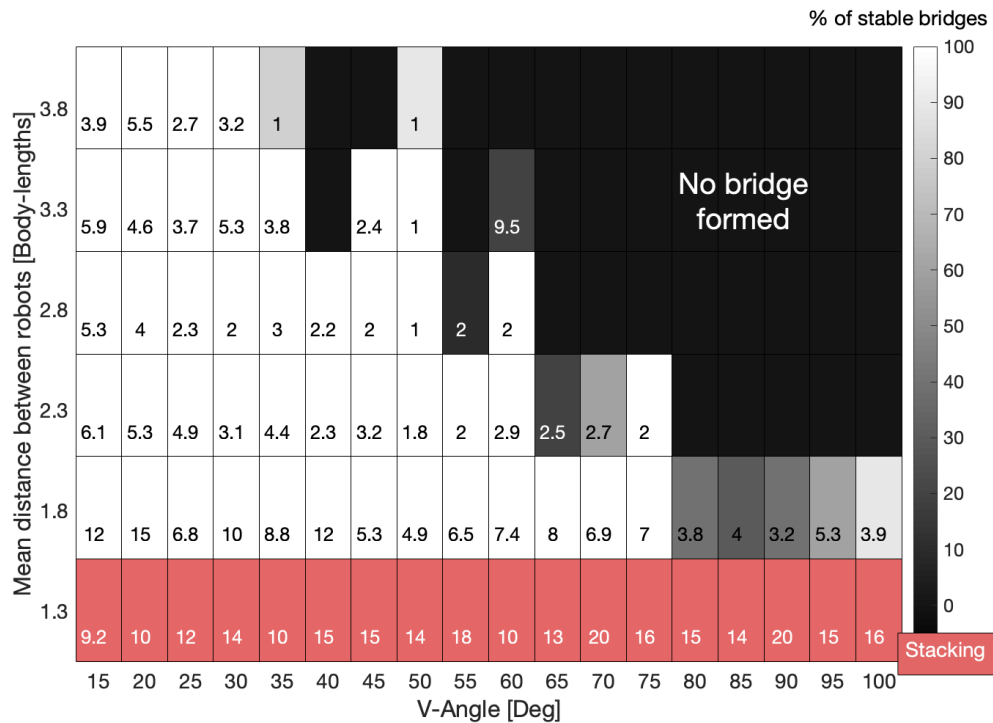
**Figure 4.8:** Bridge formation types, for multiple traffic rates

the duration of traffic. At higher traffic levels, larger and more complex bridges form. A bridge can be stable (unchanging) for a duration of time and then grow due to a burst of traffic. We consider a bridge stable if no new robot enters the bridge state for 40s, the time required for a single robot to traverse to the bottom of the longest terrain. At very high levels of traffic, unstable situations occur which we define as stacking: robots continually collide, resulting in a traffic pileup that extends out of the V gap to the terrain start. In this case, the simulation is halted and a stacking condition is recorded. Intuitively this occurs when robots are introduced faster than they can move out of the way. Robots travel approximately 0.5 BL per flip, so any slow down at 1.3 BL apart (where grippers are less than one flip from each other) creates this stack-

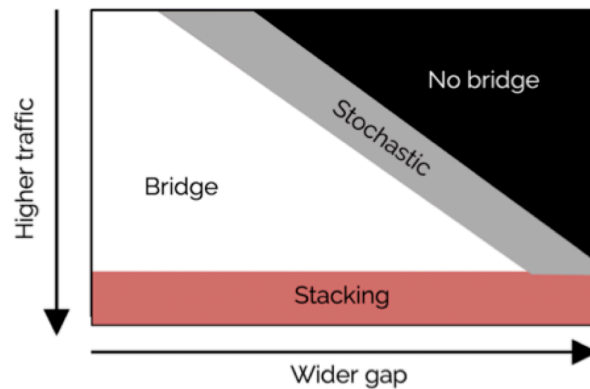
ing situation. Anecdotally, myself and others have also witnessed stacking in army ants while observing experiments in Panama. Ants can solve pile-ups by finding or forming a wider path, which the current 2D simulation does not allow.

Overall bridge formation results across all terrains and traffic rates are shown in Fig. 4.9. Each column corresponds to a V-shape angle and each row corresponds to a traffic value (mean inter-robot distance). Black sections with no number indicate that no bridge was formed, while red signifies stacking. For the remaining cells, the gray shade represents the percentage of observed stable bridges over the ten repetitions of the experiment. The number in the cells is the average number of robots in the bridge across experiments. The results match the desired adaptive behavior seen in biology: for low traffic and easy terrains, no bridges form; for high traffic and narrow terrains, bridges form, and tend to stabilize traffic; bridge size is larger if the traffic is high. At the frontier between those two areas, not all the experiments lead to a bridge formation. For very high traffic, stacking occurs (pink).

Fig. 4.10 shows the relationship between the final mean bridge height and the V-angle for different traffic levels. At very small angles, the height of the bridge is nearly at the top of the bridge, and similar for all traffic levels. As the angle increases, the difference in bridge heights increases, with some very low traffic levels (distance between robots is 3.3 or 3.8 BL) not forming bridges at all. The two dashed lines show the height for each V-angle where the span of the gap is one and two body lengths. The bridge height for different traffic levels appears to follow these trendlines, with the exception of the highest traffic level (1.3 BL) where stacking occurs in all situations. Intuitively, this makes sense; bridge formations can only occur when robots collide, and they are most likely to collide where the terrain is 1-2 BL apart. For her thesis<sup>93</sup>, Houel showed that we could significantly impact bridge building by cutting off the terrain at the two

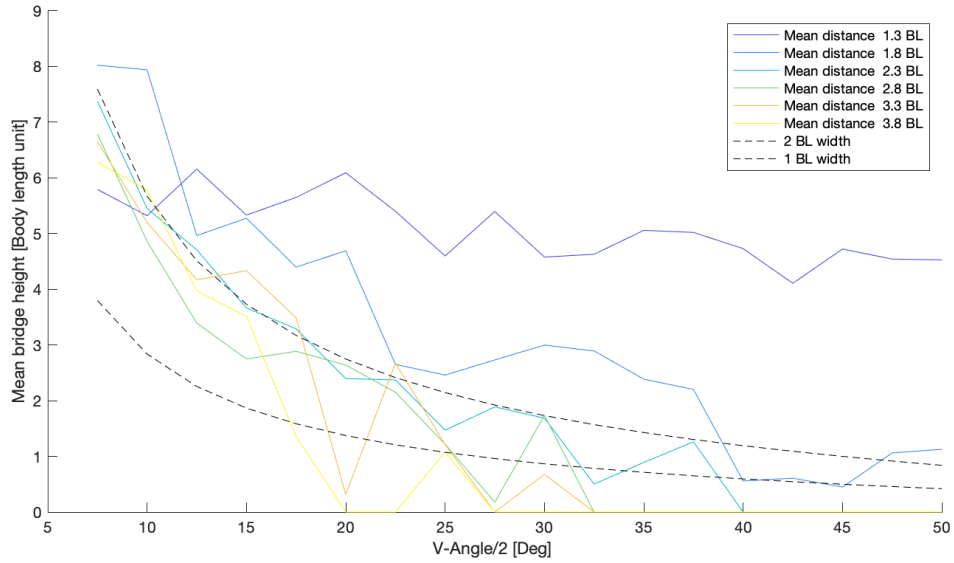


(a) Bridge formation: Each column corresponds to a V-shape angle and each row corresponds to a traffic value. Black sections with no number indicate that no bridge was formed, while red signifies stacking. Gray shade represents the percentage of observed stable bridges over the ten repetitions of the experiment. The number in the cells is the average number of robots in the bridge.



(b) General trends for bridge formation. Bridges are more likely to form in narrow gaps and at higher traffic levels. In the white and black areas, bridges either form or do not form, respectively; in a grey space between these, bridges form probabilistically. At the bottom, where traffic is high, robots will stack on top of each other, no matter the terrain.

Figure 4.9: Simulation Results: Bridge Formation

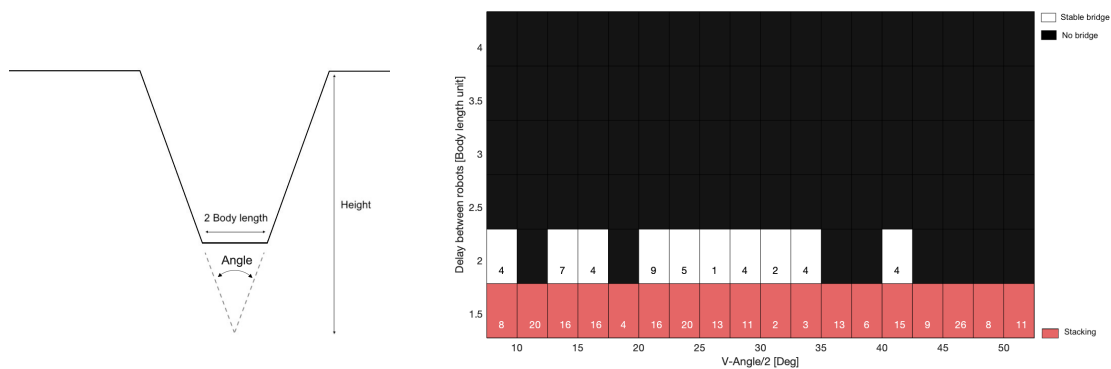


**Figure 4.10:** Mean height of bridge formation over different angles for different traffic levels.

body length point as shown in Fig. 4.11 on the left. The righthand plot shows that bridge formation was substantially reduced for this terrain, but still occurred at the higher traffic levels. Bridge building appears to function essentially as a way to smooth the terrain; which here has been done in part for the robots. We could also affect the height of the bridge by varying the standard deviation of the Gaussian distribution of traffic: larger deviations increased the height of the bridge, likely because this would create situations where robots were closer together and more likely to collide.

## BRIDGE DISSOLUTION

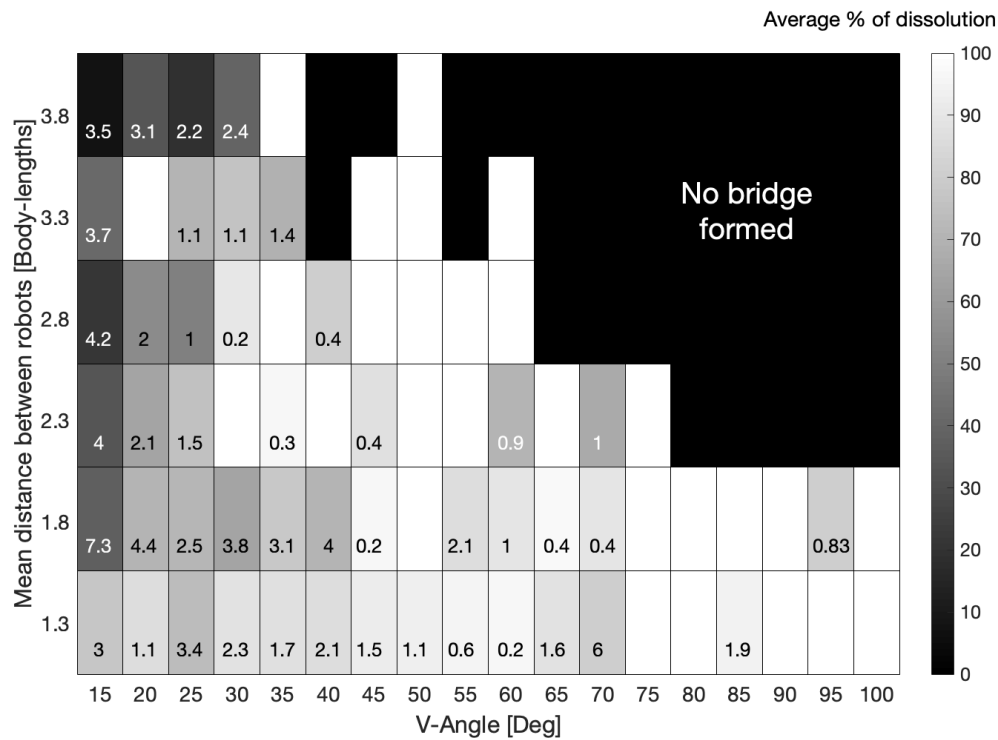
The top of Fig. 4.12 shows the results of bridge dissolution analysis for the same set of experiments as in Fig. 4.9. In the dissolution phase, the incoming traffic is stopped, so bridge robots at the top will no longer experience being grabbed by walking robots. We expect that they will



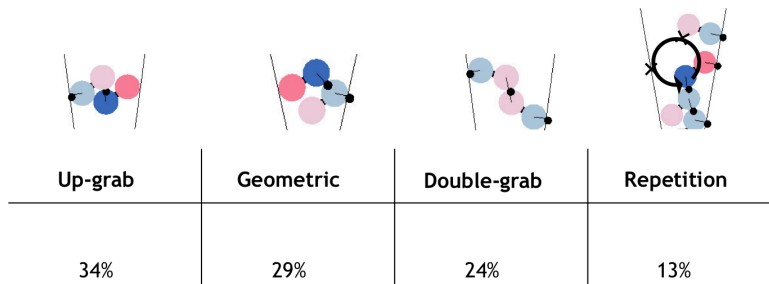
**Figure 4.11:** Cutting off the V-terrain at the point where 2 robot bodies can span the terrain as shown on the left, results in reduced bridge formation (shown on the right).

therefore transition back to flipping, exposing more bridge robots. In the results shown, most bridges completely dissolve; only 73 total did not, about 11% of the bridges formed. Of these few bridges that did not dissolve completely, the gray shade represents the percentage of the bridge which does dissolve, i.e. the percentage of bridge robots that leave after the flow of traffic ceases, averaged over ten experiments. The numbers in the gray boxes are the average number of robots stuck at the end of the dissolution step. As can be seen, when dissolution is incomplete, only a few robots usually remain stuck in the bridge structure. Even for the highest traffic situations where stacking occurs, the majority of robots are able to leave the structure, indicating that the bridge building rule may be robust even to very high levels of traffic, or if traffic comes in bursts. Bridge dissolution is the least successful at very small angles; this makes sense as in these tight spaces robots are more likely to get wedged into place or stack on top of each other. More oddly, dissolution is also less successful at the lowest traffic rate, but even here the raw numbers are low, (2-3 robots left on average); the percentage of robots left in the bridge is high because there were fewer robots in the bridge to start.

We looked carefully at these few cases where dissolution is incomplete and determined four



(a) Bridge Dissolution: The gray shade represents the percentage of the bridge which dissolves, i.e. the percentage of bridge robots that leave after the flow of traffic ceases, averaged over ten experiments. The numbers in the gray boxes are the average number of robots stuck at the end of the dissolution step.



(b) Error Conditions in dissolution and their relative percentage of the total failures. From left to right: 1. A bridge robot grabbed from below 2. Two bridge robots grabbing each other 3. Walking (top) robot stuck in the terrain 4. Trapped walking robot which moves in circles

Figure 4.12: Simulation Results - Bridge Dissolution

major classifications of errors in bridge dissolution, shown in the bottom of Fig. 4.12.

1. **Up grabs** refer to occurrences where the bottom grabs the top robot and forms a structure. The top robot cannot move because it is being grasped and has entered the bridge state, whereas the bottom robot cannot move because the top robot blocks its path. This is more likely to happen in tight gaps where the robots do not go through the neutral position. Again, as shown in Chapter 5, this is unlikely but may be possible in the robot hardware, and we suggest some possible remedies there - e.g. allowing the bottom robot to backtrack when it senses that it is “stuck.”
2. A **geometric** error occurs when a robot becomes stuck or wedged into the terrain or robot structure. While flipping, a robot may attach to another robot or terrain surface and then not be able to detach and move the next gripper. As shown in the figure, this occurs mostly during the situation where the robot has not moved through its neutral position or  $90^\circ$  of rotation. Because of this, and because the robot in simulation, unlike the robot hardware, is composed of rigid parts, the gripper is stuck due to the geometry of the terrain. We do not observe this error in the hardware system due to the built in compliance and the nature of the robot trajectory, which, as shown in Fig. 4.3 differs somewhat from the simulated robot.
3. In a **double-grab**, two robots are gripping each other. Whenever one robot tries to move, it senses the other robot gripping it, and returns to the bridge state. This also is unlikely to occur in the hardware system (though may be possible as discussed later in Chapter 5), but is an interesting case as we expect it can also occur in ants, if two ants are grabbing each other. To allow for full dissolution, the current algorithm could be mod-

ified so that agents could “shake off” other agents - by signaling their desire to leave the structure. We have observed that ants in experiments leaving the structure may become more active and struggle for a time before being able to leave, but also that this may occur for ants in the bridge that then decide to stay, indicating the ants grasping them may or may not decide to let them leave.

4. **Repetition** is essentially a variation of the Up-grab, where the bottom robot is moving, and repeats the same trajectory over and over again. The top robot is grabbed each time before it can decide to move. This may have an easier fix: by adding variation in robot speeds and decision time, we may be able to create a situation where bottom robot slows enough for the top robot to escape.

In conclusion, errors in dissolution are rare, and we believe are unlikely to occur in hardware, partly due to the compliance of the robots. Testing of additional rules and simulating in 3D may also provide some solutions for robots which get stuck.

#### 4.7 DISCUSSION AND CONCLUSION

In this Chapter, we demonstrated the use of a simple rule - “join the structure when stepped on” - to form adaptive bridge structures across V-shaped gaps, which respond to the terrain and traffic levels and dissolve when no longer needed. This is one of the first demonstrations of adaptive self-assembly in robots, and the first simulation in our knowledge where structures can change in size and shape in response to the terrain. While we focus on these V-shaped gaps, both formation and dissolution rules are not specific to the terrain and can be used to self-assemble structures in more complex terrains with multiple ramps and gaps as in Fig. 4.1 where



robots were able to form and dissolve the pictured structures.

A few exciting areas for future exploration include varying traffic levels, heterogeneous or stochastic agents, further exploration of the control rule parameters (i.e. bridge wait time), and additions or modifications to the rules. In her thesis, Houel explored changing the locomotion rules and compared the results for a pushing delay at  $90^\circ$  to one at  $30^\circ$  and  $172^\circ$ , and found that the other options increased the number of errors in dissolution substantially<sup>93</sup>. Many variations and additional rules are possible with this simulation, and varying bridge wait time in particular could help to get a better understanding of how this may affect building, which would be useful for the robot hardware.

To better understand bridge building behavior, which is affected by a large number of parameters, we limited each experiment to a set traffic level, which we cut off completely for dissolution. While we show bridges forming in high traffic and dissolving completely, ant bridges may grow or shrink if the traffic levels vary. In addition, ants (and robots) do not all travel at the same speed and may vary in how likely they are to join a bridge. Thus exploring heterogeneous and stochastic agents and varying traffic levels could provide helpful insight. In our 2D case, it is necessary for ants that are stepped on to join the bridge to prevent build up, but we can vary the bridge wait time stochastically as well as the flip wait time for variable speeds. Similarly, we can vary internal values for robot agents, such as size and mean values for flip speed or wait time - ants do not only travel at different speeds but also vary in size depending on the caste. Some ants may never choose to join the bridge if they are carrying cargo back to the nest, which could also be a useful trait for robots.

Our simulation fills the V-shaped terrain, but experiments with real army ants, ants at the bottom are able to leave the structure. While the 2D nature of our simulation prevents robots

from leaving, we could explore rules which allow them to disappear from the structure if they sense that they are no longer needed. This might be a first step towards 3D structures, where bridges will be able to adapt in more ways.

Currently self-assembled structures work to simply smooth rough terrain and reduce traffic congestion. Another interesting area for further exploration would be the creation of goal-driven structures which allow robots to reach areas that they would not be able to without the structure, e.g. cantilevers across gaps or towers (Fig. 4.1)f.

Finally, to fully understand the behavior and be able to use it in real world applications, it is essential to apply the algorithm in hardware. In the following chapter, I implement our algorithm in the robot hardware platform. The robots form and dissolve simple adaptive structures, demonstrating its potential on real world robot hardware.

## Chapter 5

*Eciton robotica:*

# Self-Assembly Behavior in Hardware

### 5.1 OVERVIEW

In this chapter, I describe the implementation of the self-assembly algorithm in hardware. In Chapter 4, I showed that simulated robots could follow a very simple rule to form structures – entering a stationary bridge state when stepped on, and exiting the stationary bridge state after a delay if no longer stepped on. This rule was sufficient to create simulated adaptive structures that form and dissolve based on traffic congestion. To implement the full algorithm in hard-

ware, I developed two fully autonomous *Eciton robotica* modules. With these robots, I was able to recreate a successful demonstration of bridge formation due to traffic congestion. To my knowledge, *E. robotica* is the only example of adaptive, amorphous assembly in a vertical plane. Only two other hardware demonstrations of adaptive self-assembly exist: Slimebot<sup>14</sup>, where a group of robots successfully reconfigure around obstacles and Swarmbots<sup>12,13</sup>, where robots form chains. *E. robotica* introduces a fully autonomous system for self-assembly that forms and disassembles structures. While the current system consists of two robots and creates very simple, single robot structures, with design improvements the hardware model holds promise for an exciting future swarm.

A key novel hardware addition in this chapter is vibration-based communication between robots. The simulated robot rule assumes that robots can detect that another robot is stepping on it or grabbing it. While it is not known how army ants detect each other, we hypothesize it is through mechanosensation, which can be difficult to implement along the length of a full robot. Many animals including many insects communicate via vibration; examples include Carpenter ants (*Camponotus*) and termites which both use vibration as alarm signals<sup>27,94,95</sup>. Inspired by these examples, I implemented a vibration-based communication, so that when a robot walks on another one, it transmits vibration pulses that travel through the body of the other robot. The other robot can then detect this pulse through the use of onboard accelerometers. While vibration is widely used in tech to communicate to humans (the vibration of a cell phone or smart watch, for example), we believe this is the first case of robot to robot communication via vibration. This could be a powerful communication mechanism for soft robots that need to coordinate physical activities, e.g. self-assembly and collective transport.

The chapter is organized as follows:

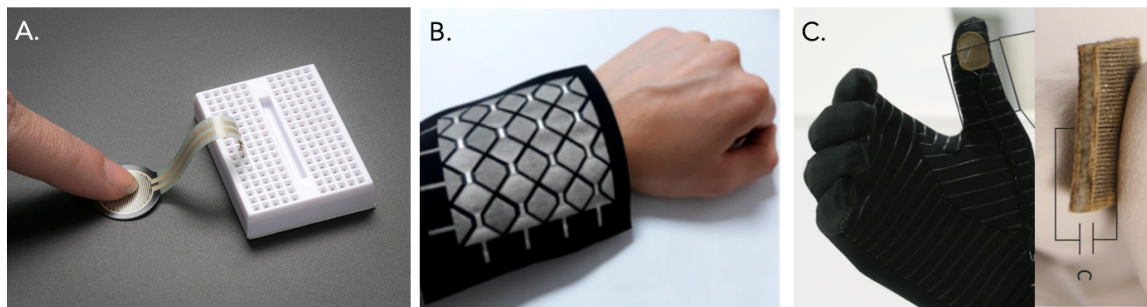
**5.2 Detection of Robot Agents** introduces and explains the detection of other robot agents through vibration pulses

**5.3 Self-Assembly Experiments** describes the experimental set-up, explains the implemented control algorithm and then describes the final experiments, and improvements for future work.

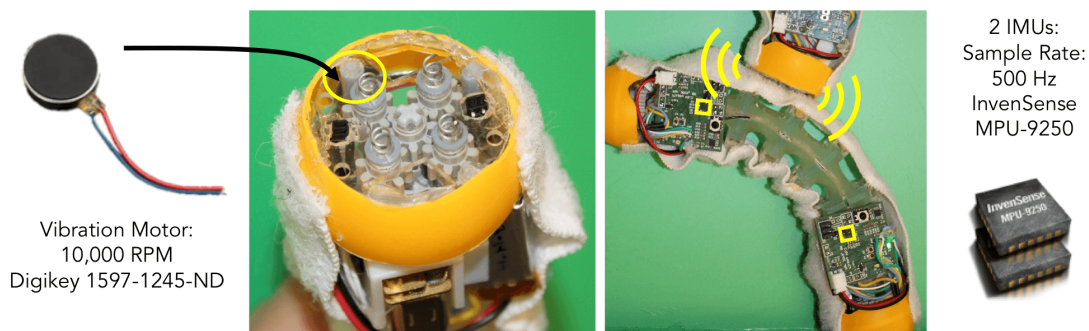
## 5.2 DETECTION OF ROBOT AGENTS

To self-assemble into structures, robots add one local control rule to the default flipping motion: if they detect that another robot is climbing over them, robots become stationary. The simulations demonstrates that this control rule allows for the formation of adaptive self-assembled structures. To achieve this in hardware, each robot must be able to detect if another robot is climbing over it, and must be able to do so no matter where the other robot has attached to it.

In previous literature, many methods have been used to detect collisions or touch. Several of these, such as mechanical switches, were discussed earlier in Chapter 3 in developing the gripper touch sensor; however, these would be unable to cover the robot body. Off-the shelf



**Figure 5.1:** A) Off the shelf pressure sensor from Adafruit B) and C) Custom touch sensors from Vu et al.<sup>96</sup> and Atalay et al.<sup>97</sup> respectively.



**Figure 5.2:** The vibration motor (right) is inserted into the robot gripper between the orange plastic cover and the acrylic gearbox casing. Adhesive on the back of the motor helps it stick to the plastic cover, securely in place. When the gripper attaches to another robot (left), the IMUs on either side can pick up the pulsing signal.

pressure sensors such as in Fig. 5.1A would need to be combined in a long array to cover the robot. Custom touch sensors like those developed by Vu et al.<sup>96</sup> or Atalay et al.<sup>97</sup> mimic the behavior of skin and are often incorporated into wearables as pictured in Fig. 5.1B and C. These custom designs require time and effort to manufacture, but could more easily cover the whole length of the robot. Similarly, layers of conductive fabric and/or thread might have been incorporated and either the pressure from the corkscrews or the corkscrews themselves could complete a circuit or act as a variable resistor. However, a central problem with both these off-the-shelf and custom pressure sensors is that they are affected by both pressure and bending. The movement of the robot module, thus could be potentially confused with contact from another robot.

Inspired by vibration systems in nature such as Carpenter ants, termites, treehoppers, and jumping spiders<sup>27,94,95</sup>, *Eciton robotica* uses a vibration motor on each gripper to send pulses and signal its presence. When one robot is attached to another robot, the vibrations travel through the soft body and are detected by the bottom robot through its IMUs. Vibration offers several advantages. 1) Pulses can be transmitted through the body, making sensing in-

dependent of attachment location. 2) It makes use of sensors (i.e. the IMU) that were already in use on the robot and 3) It has the potential for additional sensing and communication of messages. One exciting possible expansion of this research would be to use vibration to send messages via a kind of Morse code.

Pulses were chosen over continuous vibration for two reasons. One, every robot follows the same local rules, so each robot must both create vibrations and detect them. By having offset pulses, a robot can ignore its own vibrations. Secondly, vibration consumes significant power, and pulsing allows robots to conserve battery life.

The implementation of vibration consists of two parts, sending vibration pulses and detecting them. The process of detecting pulses can be broken down into 1) achieving the correct sample rate and 2) processing the signal.

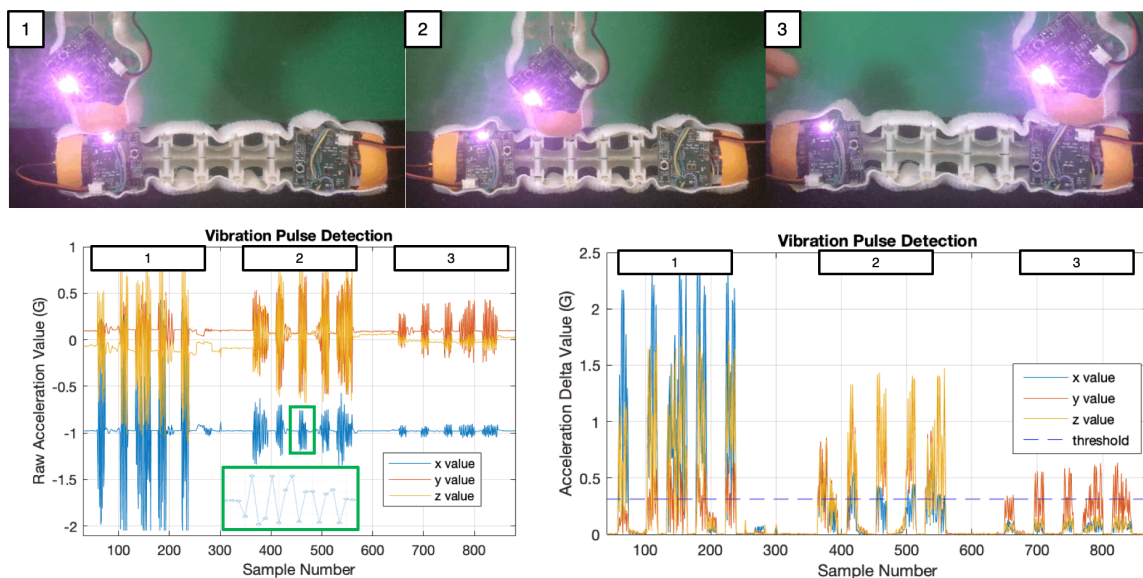
- **Sending pulses:** Sending pulses is fairly simple. The vibration motor is inserted into the gripper as shown in Fig. 5.2. Pulses are sent at a set interval rate during each state (flipping, detaching, attaching), at least 10 times per flipping cycle (the total number will depend on the length of the attaching and detaching phases). Each pulse lasts for 200 ms.
- **Detecting pulses:**
  - *Sampling:* The vibration motor (Digikey 1597-1245-ND) runs at 10,000 RPM or approximately 166 Hz, making the minimum sampling or Nyquist rate 332 Hz. The maximum sample rate of the MPU-9250 is 1 kHz; I ran it at 500 Hz for improved power consumption while still meeting the minimum requirements. As shown in Fig. 5.3 in the zoomed in section, the IMU receives about three samples

per cycle of the vibration motor, as expected for the given frequencies. When sensing for a pulse, the robot takes a set of fifteen samples at a time, to ensure that it achieves the required sampling rate and is not delayed by other control processes.

- *Processing* For each of the 15 samples taken, the robot calculates the difference between them and records this as a delta value. By using the change in values, robots can easily distinguish a vibration pulse from the changes in acceleration values due to gravity and movement of the robot, essentially creating a computationally cheap high pass filter. In initial testing, robot determines that it is “stepped on” and enters bridge state IF at least 3 or more of these delta values is above the threshold. For the final self-assembly demonstrations, this was increased to 7. Delta values and the threshold are shown in Fig. 5.3 (bottom right). The final control process is also shown in more detail in Fig. 5.8 as the `detectPulse()` function.

As a first test of the vibration detection, I conducted some simple tests. (a) I put one (detector) robot flat, and then touched the gripper of the other (transmitter) robot to different points along the body of the detector robot, as shown in Fig. 5.3. Pulses were sent five times at each location, for a total of 15 tests, all of which were successfully detected. I was also able to induce detection of a “pulse” by hitting the robot grippers. This proved to be useful in some cases, as I could induce a bridge state. However, we did not want the robot to interpret its own flipping motion or the impulse from encountering a new surface as a robot pulse. I therefore also tested detection in a single robot while it conducted its normal flipping motion, using the same detection function. Over 20 flips, no false positive detections were recorded. Therefore I purposefully kept the threshold value low to reduce the number of false negatives. False negatives may be more detrimental to bridge building behavior than false positives, as false positives will





**Figure 5.3: Top:** In this experiment, a “detector” robot was placed flat on the ground and uses the accelerometer on the lefthand PCB only to detect pulses. A second “pulser” robot was held such that its gripper touches the detector robot at different locations along the body (left, middle, right). The **Bottom Left** panel shows the raw accelerometer values. The values decrease with distance from the detector robot’s left gripper. However, they are still quite clear and distinguishable from the times when no pulse is being sent. The zoomed in section in green shows the data points distinctly; the sample rate is approximately three samples per cycle. The **Bottom Right** panel shows the same data but plots the delta values, i.e. change in accelerometer values. By using the change in values, robots can easily distinguish a vibration pulse from the acceleration values due to gravity. E.g. x values in the raw data plot center around -1G, but this will change as the robot flips in different orientations. The chosen threshold change in delta values is shown in dashed blue. For clarity I have normalized the data plots to 1 G, but all robot code uses the raw, unconverted values, where 16000 is equivalent to 1G).



**Figure 5.4:** An *E. robotica* robot climbs over a second robot, which detects its presence as indicated with pink LEDs.

simply encourage the robots to stop more frequently.

As a final proof of concept, I tested detection of a robot climbing over another robot. Fig. 5.4 shows video frames of one robot climbing across the body of a second robot. Each time the climbing robot makes contact and attaches, it sends a vibration pulse, indicated by flashing pink LEDs. The bottom detector robot detects that it is being walked over, and likewise uses pink LEDs to indicate that it has detected the vibration pulse of the top robot. This was done seven times, six in the bottom left corner of the box (as shown in the Fig. 5.3) and one in the bottom right, for a total of 30/30 true positive detection cases of the top robot. No false negatives were recorded, and only 3 false positives, all of which were cases where the robot was actually on top of the other robot (2 cases where detection was caused by collision of the top robot with either the bottom robot or the ground, and one where detection appeared to occur, which may be a human recording error).

As in Chapter 3, climbing over other robots was still a struggle for the *E. robotica* modules; 1 out of 7 attempts was completely successful without human intervention, though the flip success rate was much higher. Even this one success shows that a fully-autonomous and untethered *E. robotica* robot is able to climb over a soft, robot-made structure, and the robot inside the structure can consistently sense the presence of the moving agent. These are all of the ingredients necessary to create ant-inspired self-assembled structures, which I demonstrate in the following sections.

### 5.3 SELF-ASSEMBLY EXPERIMENTS

This section describes the hardware demonstration of the full self-assembly algorithm. As a reminder, robots follow a simple state machine to move consisting of flipping, attaching and

detaching. To form structures, the robot adds an additional “bridge” state: if it senses that another robot is attached to it, the robot waits in place until it no longer senses the other robot. In simulation, we ran experiments over a wide range of terrains and bridges with large numbers of robots, to show that this simple rule leads to adaptive structure formation.

Although the control rule is simple, there is always a significant gap between theory and simulation, and a real-world implementation in hardware with full autonomy and no human intervention. This gap can be even larger for soft robots, where simulations do not accurately capture the complex body characteristics. Robots in hardware also face challenges that robots in simulation do not.

Our goal for the hardware demonstration was to test a complete version of the algorithmic concept and better understand the gaps between theory and practice. To this end, I developed two fully autonomous *E. robotica* robots. **With these two prototypes and our algorithm, I was able to successfully demonstrate simple bridge formation and dissolution, induced by traffic collisions in a simple terrain.** Based on extensive experiments, I also discuss improvements to the design to make self-assembly more reliable and enable expansion to a larger swarm. My final demonstrations show the promise of soft self-assembly, and while limited, they push the boundaries of what has been accomplished in the field so far with soft robots and adaptive self-assembly.

### 5.3.1 SELF-ASSEMBLY ALGORITHM

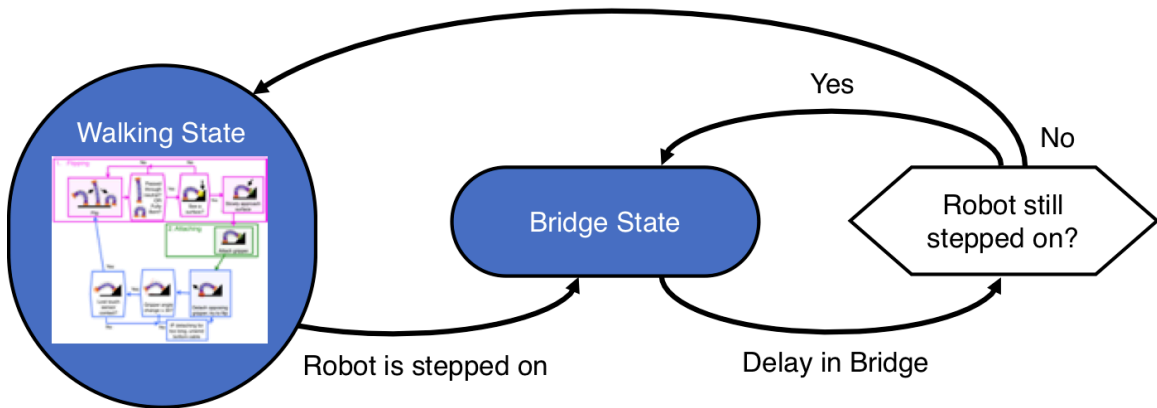
As in simulation, the *Eciton robotica* robot in hardware has a *Walking* state and a *Bridge* state. The three state machine introduced in Chapter 3 is folded within the *Walking* state as sub-states (Fig. 5.5). Robots enter the bridge state when they sense they have been touched by an-

other robot and exit if they have not felt another robot within a certain wait period.

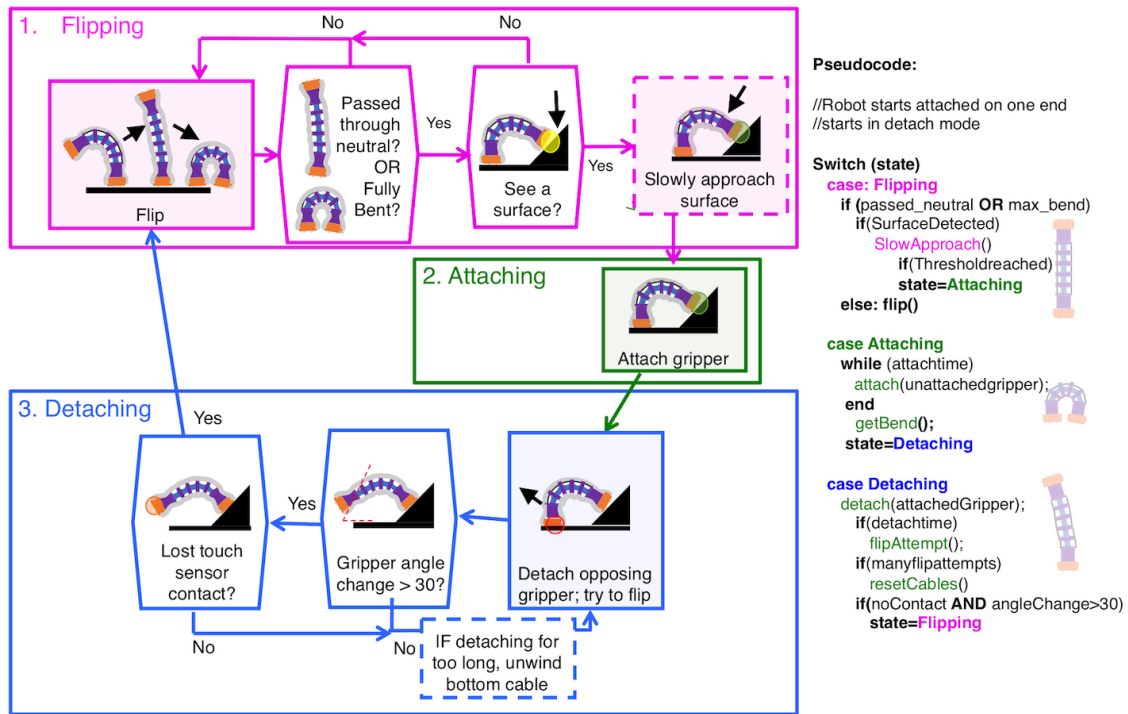
The three states for locomotion (flipping, attaching and detaching) is shown earlier in Fig. 3.23 and again in Fig. 5.6 (flipping is purple, attaching is green, and detaching is blue). These remain mostly the same. To improve reliability and error tolerance from Section 3.6, I added additional safeguards as shown in Fig. 5.6 in dashed lines; I allowed the robot to attach if fully bent, added a slowed approach to improve attachment, and added a cable reset in the detach state, the motivation for which is discussed more in 3.6. These improved the robot performance, although full feedback control within the attachment state and flip state could further improve error tolerance as discussed later in this chapter.

Fig. 5.7 gives an overall view of the control layers and shows in more detail how the locomotion control is folded within the state machine of the robot. Throughout each of these local locomotion states, the robot sends pulses, to indicate its presence to any listening robots. It in turn listens to see if any pulses are detected and if so, enters the Bridge state.

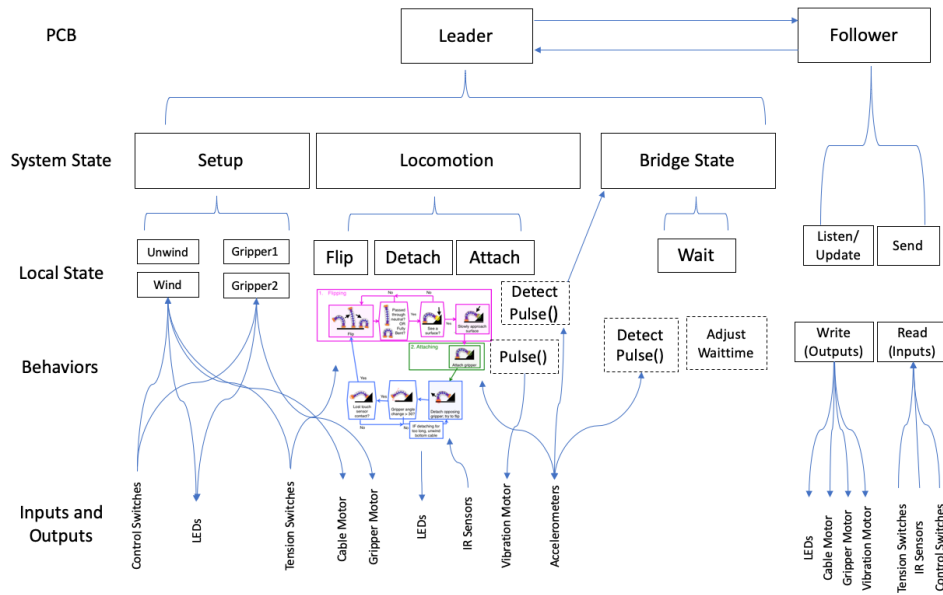
Fig. 5.8 shows pulse detection and the Bridge state in more detail. While the rule “stepped when stepped on” is simple, in practice it has several tunable parameters - i.e. how to signal the presence of other robots (pulse duration and frequency of pulses), how to determine when the robot is stepped on (pulse threshold for change in acceleration and the threshold for number of samples), and how that signal is converted into a wait time in the Bridge state. Pulse signaling and detection were discussed in the previous section 5.2, and the final values were tuned during experiments. The conversion of the detection into a wait time is a function  $H(n, p)$  where  $n$  is the number of samples detected during a pulse, and  $p$  is the number of previous pulses. Currently, wait time is added to the robot as  $C + Gn$  where  $C$  a constant value, added each time a pulse is detected and  $G$  is a gain which increases the wait time with the number of samples



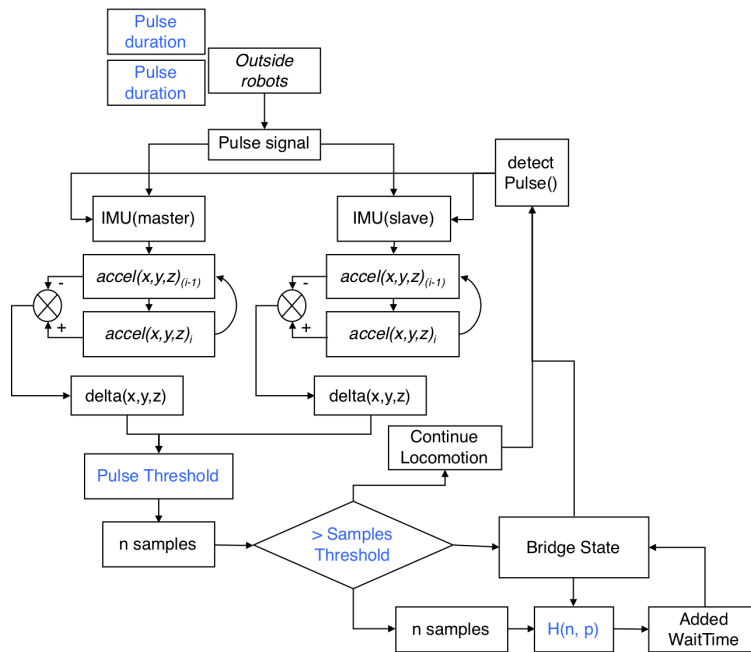
**Figure 5.5: Self-Assembly Algorithm:** As in simulation, the *Eciton robotica* robot has a *Walking* state and a *Bridge* state. The three state machine introduced in Chapter 3 is folded within the *Walking* state. Robots enter the *Bridge* state when they sense they have been touched by another robot and exit if they have not felt another robot within a certain wait period.



**Figure 5.6: Locomotion Control:** *Eciton robotica* robots use the state machine introduced in Chapter 3, with a few modifications for improved reliability and error tolerance. The flip state is shown in purple, attaching in green, and detaching in blue. Modifications are dashed.



**Figure 5.7:** System Control Layering for *E. robotica* robots. System states such as Set-up and Locomotion have local sub-states which perform various behaviors to interact with robot inputs and outputs. Pulse detection is run as an interrupt in the overall locomotion system state, so that the robot can quickly move to bridge state if a pulse is detected. All control is handled by the leader board; the follower board simply updates outputs and inputs as needed.



**Figure 5.8:** Control for Bridge State and Pulse Detection. Tunable parameters and control are highlighted in blue.

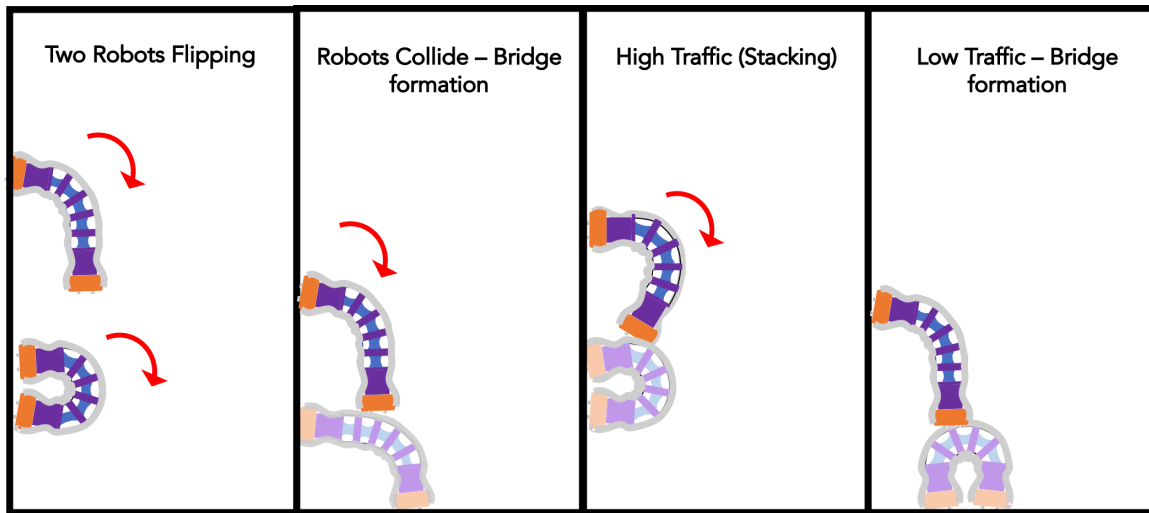
detected. This allows an increased wait time for stronger signals, while decreasing the effect of possible false positive detections. As discussed later, adding a  $p$  term where this wait time increases more for each pulse detected may increase reliability of the robot and has a basis in biology, as local traffic is exponentially related to time spent in self-assembled army ant bridges<sup>3</sup>. The given  $H(n)$ , however, still allows us to form structures, as described in the following sections.

### 5.3.2 EXPERIMENT SET-UP

#### TERRAIN SELECTION

In simulation, we used a V-shaped terrain based on army ant structures to test bridge formation and dissolution. In the hardware experiments, I simplified the terrain further, using the box-shaped track from previous testing. Each corner serves as a  $90^\circ$  V.

Within the box, four corners provide four choices of possible V shapes. Results from climbing tests discussed in Section 3.6 suggest that the current prototype is most reliable when climbing downwards and on flat surfaces, and when climbing over robots in the lower left (down to flat) corner (Corner D); I chose to focus experiments on this terrain. As shown in Fig. 5.9, robots start climbing downwards, and, if the spacing is close enough, they will collide to form a bridge in the corner transition. Depending on spacing, they may also collide before or after the corner, forming a temporary block on the vertical or horizontal plane. Early collision occurs when traffic levels are high (i.e. robots start closer together), and is similar to the stacking situation observed earlier in simulation; late collision occurs when robots are spaced farther apart (i.e. low traffic). This is also observed in simulations, where bridge robots can form along the inclines of the V.



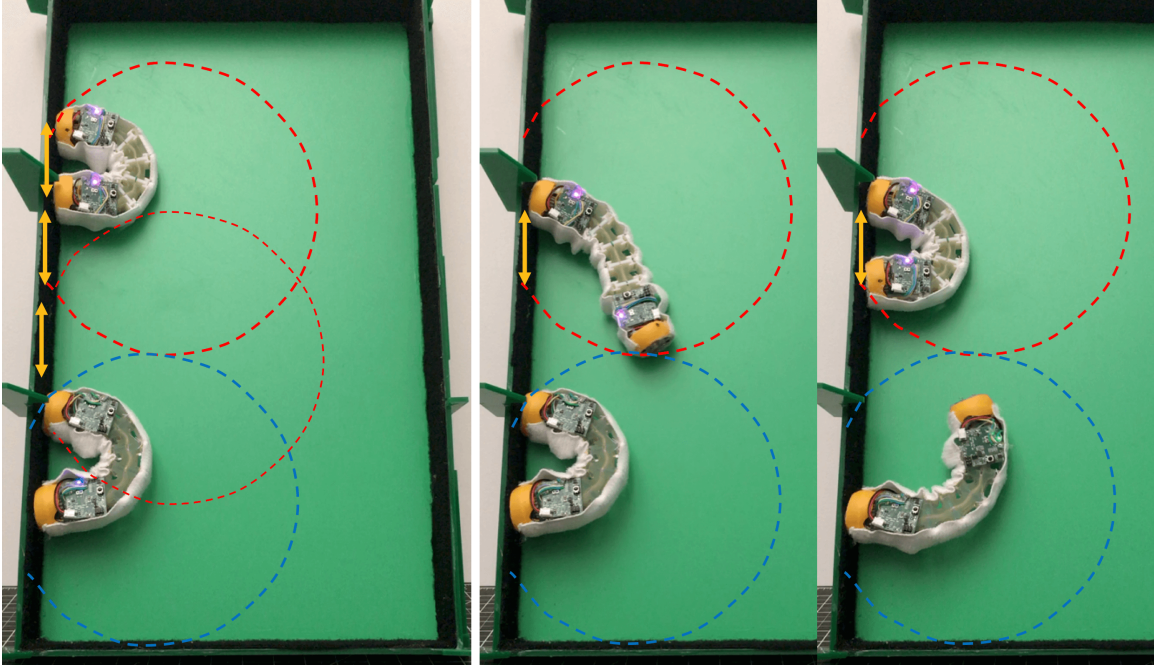
**Figure 5.9:** Bridge Formation. From left to right. A) Two robots are flipping down a vertical surface. B) The top robot collides with the bottom robot, causing a bridge to form. C) Here traffic is high (robots are close together) and the robots collide early, causing bridge formation along the flat vertical surface. This is equivalent to stacking in simulation, but may occur more frequently in hardware because the speed of travel varies for each robot. D) Robots are farther apart (low traffic) and collision occurs later, but still causes a bridge to form.

Fig. 5.10 shows the initial set-up of the hardware experiments, using the same box platform as our previous climbing tests in Chapter 3. The majority of tests were conducted with the box covered. A few, including the bridge formation and dissolution shown in Figures 5.10, 5.12, and 5.13 were conducted with the cover removed to reduce glare. In these cases, the box was set at a slight angle, approximately  $75^\circ$ ; this encourages the robots to lean slightly on the back wall and, more importantly, to stay on the track. This did not appear to have any significant effect on climbing, though the robot was more likely stray off of the track with the open setup.

#### ROBOT SPACING (TRAFFIC LEVELS)

As in the simulation results and in army ant bridges, robot traffic levels determine whether robots form structures. The spacing and phase between robots in set-up is thus critical. The hardware set-up here is similar to the simulated V-shaped terrain with a  $90^\circ$  angle. At this wide





**Figure 5.10:** Hardware set-up for assembling structures: The left shows the initial starting positions of the robots, with their flipping trajectories in the dashed lines. Spacing is approximately two robot lengths, shown in yellow. As seen on the left, robots are set here to be out of phase.

angle in simulation, robots formed bridges only when they were very close, i.e. when the center to center distance was 1.8 body lengths or less. As in Fig. 5.10, to successfully form a structure, robots were placed approximately two flip distances away from each other. The center to center distance of the robot grippers is shown in yellow, though this may vary; here the bottom robot is taking a much larger step. The dashed lines show the flipping trajectory of the robot: red for the top robot and blue for the bottom.

Although it will take the top robot two flips to reach the first position of the bottom robot, the trajectories for the two robots on their next flips are touching and may overlap; thus phase is also important to prevent unwanted collisions during the flipping motion. As shown in the two following panels, the robots are out of phase with each other; one flips while the other is

attaching and detaching. The projected trajectories of the robot also show the potential for bridge formation. The second flip for the top robot, shown in a lighter dashed red, intersects with the bottom robot's bottom gripper. We can predict therefore that the top robot will collide with the bottom robot during its second flip, and they will form a structure as shown later shots in Fig. 5.12.

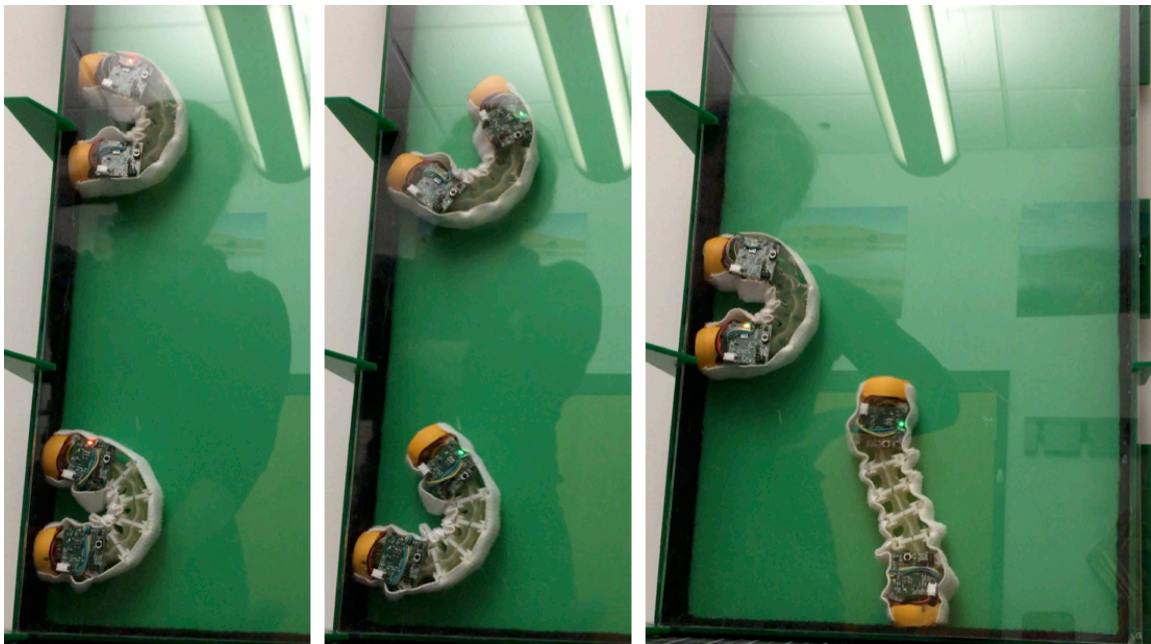
Fig. 5.11 shows what happens when robots are spaced too far apart in set-up. Robot spacing is farther apart, approximately 3 body-lengths, and robots are in phase as shown in the left and center pictures. The distance between robots is too large, so the robots do not collide and form structures but simply continue around the box.

These examples show how **timing is essential in forming bridges** due to the dependence on traffic levels. However, timing is also more complex in hardware. In simulation, robots were completely homogeneous and moved at constant speeds. Attaching and detaching were instantaneous. In contrast, *E. robotica* **robots in hardware do not travel at a constant speed**. Attachment and detachment are not instantaneous - in fact, these states take more time than the flip cycle - and the periods are variable due to the nature of the feedback control. Even the period of the flipping state may vary because the start and stop points may change. If a robot takes longer in the detaching state, it may need less time to flip, since detaching involves trying to flip in order to shake the corkscrews free. The flip distance, as can be seen in Fig. 5.10 is also not always consistent, even when traveling along a flat surface. The robot has a range of distances where it can attach; the grippers may be very close as shown on the top robot or farther away as on the bottom.

Due to these variations in speed, and small differences between robots in sensing and fabrication **the system in hardware is not completely homogeneous**. This makes it less predictable

than in our simulations. The simulator also allowed us to use many robots at many traffic levels and terrains, conducting a large parameter sweep to observe where and how bridge formation occurred. In contrast, for the demonstration in hardware, we are limited to two robots and care about conditions where bridges *do* form. In order to test our algorithm, robots must collide.

To create a more predictable scenario and increase the likelihood of bridge formation, I deliberately created further non-homogeneity between the robots, giving the bottom robot a short delay between the attachment and detachment states, and making it slower than the top robot. This also increases the likelihood that collisions will occur when the robot is attached to the surface at both ends, which is advantageous for more stable structures. The delay is also analogous for the delay between flips in simulation. The length of the delay is the same as the length for a single bridge state.



**Figure 5.11:** Robots at low traffic levels do not form bridges. Here robot spacing is farther apart, approximately 3 body-lengths and robots are in phase as shown in the left and center pictures. The distance between robots is too far, so the robots do not collide and form structures but simply continue around the box.





**Figure 5.12:** Robot Bridge Formation: The bottom robot senses the top robot and goes into bridge state while the top robot walks over it.



**Figure 5.13:** Robot Bridge Dissolution: The bottom robot senses when it is no longer being stepped on and continues to move.

### 5.3.3 RESULTS

I ran 26 experiments in order to form structures with two robots. An overview is given below and in the following page on Fig. 5.14:

- In 9 of these experiments, no contact was made between the robots, either due to individual robot errors, or due to spacing between the robots (i.e. traffic was too low). Fig. 5.11 shows one case where traffic is too low.
- In 14/17 of the remaining bridge experiments, the bottom robot both sensed the top robot and entered into bridge state successfully.
- However, in only 1/14 formed structures was able to successfully form *and* dissolve; this case is shown in Fig. 5.12 and 5.13 respectively. The most common error was early exit of the bridge robot, discussed more in the following section.
- In 3/17 of the remaining bridge experiments, the robots made contact but with the bodies of the robot and not the gripper, so the bottom robot could not detect the top robot.

Fig. 5.12 and Fig. 5.13 show the first fully successful example of both bridge formation and dissolution respectively. In Fig. 5.12, the top robot climbs onto the bottom robot that is in the corner. The bottom robot goes into bridge state and the top robot continues to walk over. In Fig. 5.13, after the bridge robot has sensed that no robots have stepped on it for the preset bridge delay time, it exits the bridge state and continues to move.

In the rest of this section, I discuss successful cases and the cases where this process failed and why in detail.

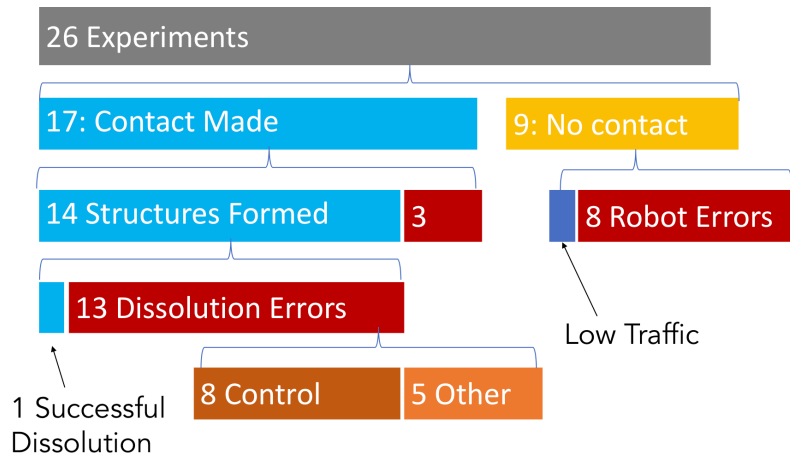


Figure 5.14: Self-Assembly in Hardware: Results Summary

#### 5.3.4 DISCUSSION

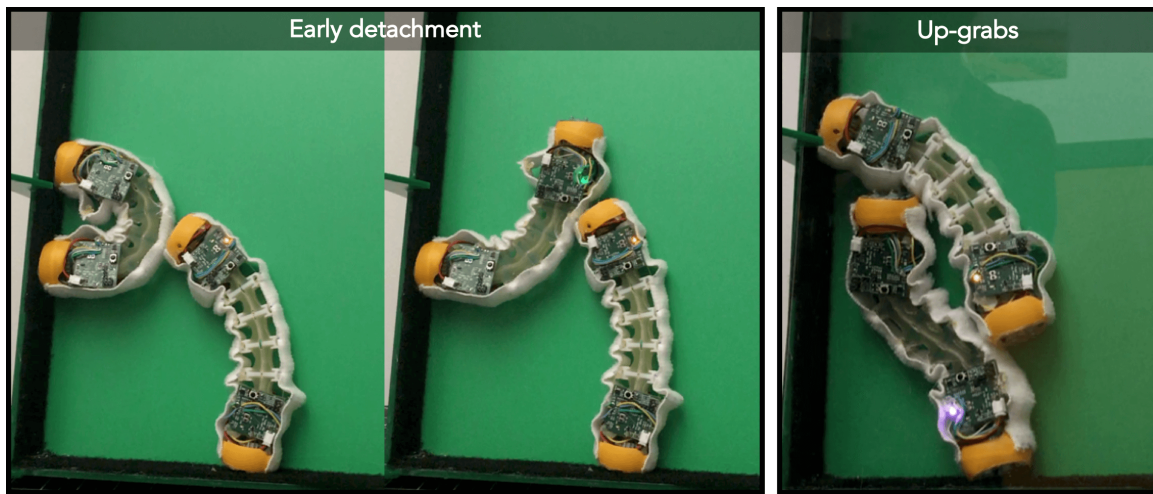
Overall, failures while climbing downwards or over another robot significantly affected many of the experiments, in some cases causing the experiment to fail before contact (8/9 cases of no contact as seen above) and in one cases causing failure after structure formation (1/13 of dissolution errors). Robots leaving the track when making contact with another robot also limited experiments and was the source of error in 3/13 cases after structure formation. Together with one battery error, these account for the 5 “Other” errors shown in Fig. 5.14.

Despite these errors, **the implementation of the first stage of the self-assembly process, structure formation, was successful**; in 14/17 experiments where contact was made, the top robot made proper contact with the lower robot, and the lower robot successfully detected and transitioned to bridge state. In two of these cases, the bridge robot spanned the corner D. In 11 cases the bridge robot formed early on the vertical wall (as shown in Fig. 5.9C) and once it formed on the flat surface (Fig. 5.9 D). The preponderance of early bridges not surprising, since we have purposefully created high traffic levels with tight robot spacing in order to ensure

contact and bridge formation.

While bridge formation was effective, the robot hardware stumbled in bridge solution, dissolving the bridge successfully in only one case. **In the 13 cases where robot bridges formed but were not able to successfully dissolve, 8 of the errors were due to control parameters.** The most common error (7 occurrences) was early detachment of the bottom robot, shown in 5.15 on the left. In simulation, robots were always aware if another robot was attached to them. However, in hardware, the robot is only aware that another robot is attached to it if that robot is currently sending a pulse signal. Therefore the frequency of the pulsing behavior and the wait time in the bridge state are important and somewhat dependent parameters; the time between pulses must at minimum be less than the wait time in the bridge state. Increasing wait time in the bridge state may make for more reliable bridge dissolution. In addition, we may consider slight modifications to our rules. Garnier et al.<sup>2</sup> shows that the time spent in the bridge is exponentially related to the local traffic, whereas the *E. robotica* algorithm is simply proportional - each time the robot senses another robot on top of it, it adds a set wait period to its time in the bridge. Adding a multiplier or otherwise introducing longer periods for additional robots sensed may make the algorithm more robust both to false positives (the robot will attempt to leave the bridge state more quickly) and false negatives (previous sensing of the top robots will compensate for the bridge robot failing to sense the top robot).

The last of the 8 Dissolution Control errors was related to early detachment, but categorized separately because it occurs due to timing and cannot be fixed by simply adding a longer bridge delay. This case, an “Up-grab,” due to its similarity to upgrabs in simulation, is shown in Fig. 5.15 on the right. Here the bottom robot has already detached from the surface at the point when it is stepped on. The two robots end up stuck, each one preventing the movement of the



**Figure 5.15:** Common Errors in Self-Assembly Experiments. Left: Early Detachment of the bridge robot before the moving robot has had a chance to fully detach and move on. Right: “Up-grabs,” the top robot contacts the bottom robot after it has already detached. Both robots are in the flip phase and trying to move. Eventually in this position, the bottom robot may be able to grab the top robot and force it to enter the bridge state; however, because it is grabbing “up” both robots will be stuck.

other. While the bottom robot eventually is able to sense the top robot, and enters the bridge state, human intervention is needed to prevent the top robot from overextending and breaking the tension switches.

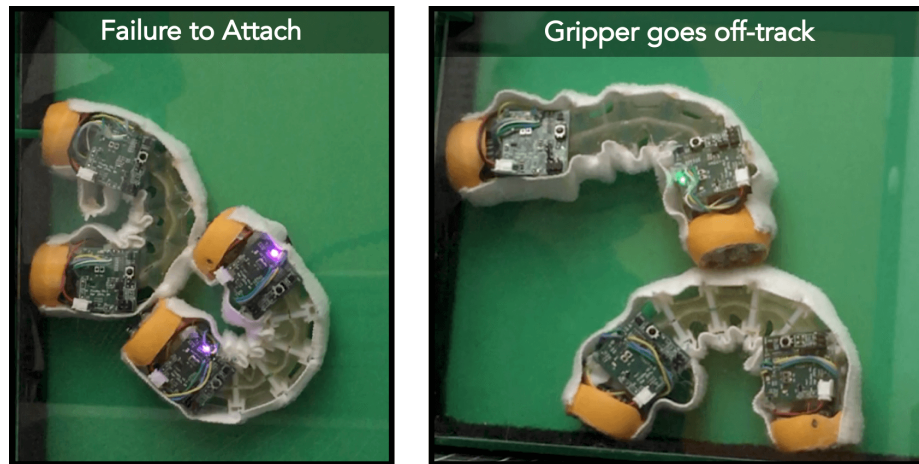
Two possible rule changes could address early detachment and timing:

1. If the [bottom] robot senses another robot attach to it while detaching, it should reattach. This will ensure a stronger attachment point while the top robot moves along the structure.
2. If the [bottom] robot senses another robot attaching while it is moving, it should reverse direction and reattach before entering the bridge state.

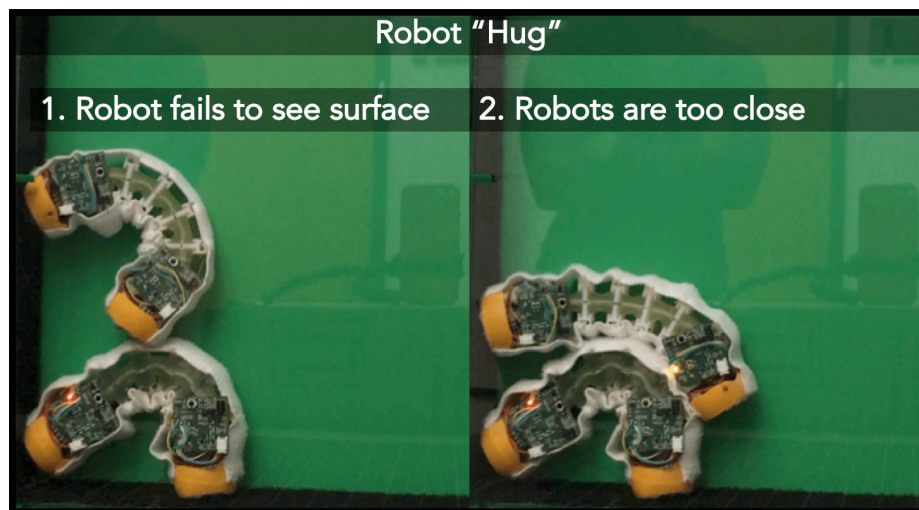
For the “Other” Dissolution Errors (Fig. 5.14), Fig. 5.16 on the following page shows attachment and track errors. In the first, the robot fails to sense surface to attach, possibly because it



is simply still too far. In another case, shown on the right, the gripper goes off the track (in this case another robot), tilting sideways. In both of these cases, the gripper will sometimes become stuck on the rigid pieces of the bottom robot, and the robot will not be able to bend further.



**Figure 5.16:** Common Errors in Self-Assembly Experiments. Left: Robot appears to be close enough to attach but does not see the surface. Right: Gripper slips and tilts off the robot, and thus cannot see a surface to attach to.



**Figure 5.17:** Common Errors in Self-Assembly Experiments. Robot "hug" error occurs in two steps. 1. The top robot makes contact, but fails to see the surface and slips past it (shown here) and will then attach very low, close to the other robot. 2. Since the robots are already very close, the top robot will make contact with its body and not its gripper, making it unable to attach and preventing either robot from moving.

Finally, in 3 cases out of the 17 where contact was made (Fig. 5.14), no bridge formed. In one of these cases, contact was simply too brief. Fig. 5.17 on the previous page shows another interesting case where contact is made along the body of the robot rather than the gripper, which prevents the pulse from being sent and received. This error occurs when the top collides earlier with the bottom robot, but fails to sense the surface and slips past it. The gripper then attaches and is too close to the bottom robot, becoming stuck and wrapped around it.

### 5.3.5 CONCLUSION AND FUTURE IMPROVEMENTS

With *Eciton robotica*, I successfully demonstrated the formation and dissolution of adaptive bridge structures in hardware, following a local rule. Bridge formation occurred at a high success rate; bridge dissolution was successful, but requires further tuning of algorithmic control parameters to improve reliability.

In a final assessment, *attachment is at the root of almost all errors in the system which are not algorithmic* (i.e. early detachment and failure to dissolve). Errors were categorized as due to attachment any time the robot was unable to make a stable connection - whether the robot failed to sense a surface and became stuck or the connection was simply poor. Attachment errors often prevented the robots from coming into contact, limiting the experiments. Additionally, they account for errors such as the “robot hug” and other failures during contact. Adding all types of attachment errors, they account for 11 out of 22 total errors encountered or 50%. Improving attachment through improved manufacturing techniques, calibration, and feedback control is key to improving the reliability of the robot for future testing.

A second area for improvement is error tolerance in the control algorithm for locomotion. It’s also important to note that some attachment errors discussed above i.e. failure to sense

a surface, are fatal to the robot: the robot flips until it senses a surface but physically cannot flip past a certain point. The current control does not account for this, but allows the robot to keep tightening the cable. Although most parts of the robots are fairly robust, the tension feedback switches can easily break if cables are overtightened, for example when the robot is fully bent but does not sense a surface to attach to, and when the robot is stuck but trying to flip (e.g. robot hugs). In this case, we can add a hard stop and allowing the robot to “try, try again:” when the robot senses that it has reached its fully bent position, when it cannot bend any further, and does not see a surface to attach to, it should unwind, and try again. A similar, more general strategy could be used if the robot senses that it has not changed positions while trying to flip (e.g. in the upgrab or hug positions).

Finally, despite the need for some improvements, the system shows promise, and expanding to additional structures with more robots is an exciting future research prospect. Four or more robots would allow us to capture more complex and interesting structures. Additional terrains, including a V-shape similar to the work in simulation or the other three corners of the box could be tried, especially with improved attachment since, as discussed in section 3.6 the robot struggles with upside down and vertical terrain mostly due to poor attachment.

Overall, the *Eciton robotica* platform demonstrates the power of a simple local rule and the potential for soft robots in self-assembly. While current experiments are limited, we feel confident that the overall approach is viable and that the main hurdles are improving the reliability of the robot and optimizing the algorithm parameters.

# Chapter 6

## Conclusion

### 6.1 CONTRIBUTIONS

Army ants create incredible self-assembled structures, including bridges which can adapt to the environment and the traffic level. Self-assembly in robotics has many potential applications, but so far most of these have worked to form pre-determined shapes. Though mostly unexplored, adaptive self-assembly, as in ants, could be especially useful in emergency conditions or unknown environments where robots must react quickly to new situations.

In this thesis, I describe the design and implementation of the *Eciton robotica* platform, a soft robotic system for adaptive self-assembly inspired by army ants. This project furthers our

understanding of army ants in nature, by testing a local control rule for that may partially account for their bridge building behavior. Furthermore, this work pushes the boundaries of what has been accomplished in robotic systems for self-assembly: in my knowledge *E. robotica* is the first *soft* autonomous, mobile robot system for self-assembly, and the first case of adaptive self-assembly in a vertical plane.

My work developed both the hardware platform and the control rules of the robot system. The final modules for the hardware platform consist of flexible bipeds which use a flipping gait to climb. They are one of the first soft, flexible climbing robots, and one of only two that are autonomous and untethered. Modules climb over each other using a novel corkscrew gripper, attaching anywhere on the bodies of the other robots, which are covered in a soft, stretchy Velcro. This allows for amorphous structures, which are rare in robotic self-assembly but abundant in nature.

To build self-assembled structures, the robots follow a simple, bio-inspired, local control rule - robots stop and join a structure when stepped on. In simulation, robots using this rule can form structures in response to traffic and terrain, and dissolve them when they are no longer needed. Structure size and shape varies with traffic levels and the terrain. This simulation work was developed with help from two undergraduate students and implemented by Lucie Houel for her Master's thesis under my supervision.

I built two fully autonomous robots and demonstrated the assembly and disassembly of an adaptive structure using our simple rule. Robots sense if they are stepped on through a vibration pulse communication, inspired by ants and other insects in nature, another novel contribution. The current system is limited to simple structures with two robots, but with some improvements in hardware and optimization of the control parameters, we are confident that

this system can create more complex structures with additional robots. In addition, the local rules may be applied to new systems in 3D. I outline how this may be accomplished and some of the remaining challenges in the following section.

In sum, this thesis expands previous work on robotic self-assembly through the use of soft robots and new algorithms for adaptive structures. In robotics overall, the project introduces a novel gripper for fabric based surfaces and a communication system based on vibration pulses with potential for expansion in future applications.

## 6.2 FUTURE WORK

In the following sections, I outline possibilities for future expansions including scaling up the *E. robotica* system, testing new rules in simulation, and expanding to 3D.

### 6.2.1 SCALING UP TO SWARMS FOR ADAPTIVE SELF-ASSEMBLY

In this thesis, I built two robots and demonstrated single robot structures. This could be useful in real applications - allowing robots to work together - however, most applications would require much larger swarms and structures. While one of the benefits of swarms is error tolerance - the large groups can succeed when individuals fail - **robots must meet some minimum reliability requirements before we can scale up.**

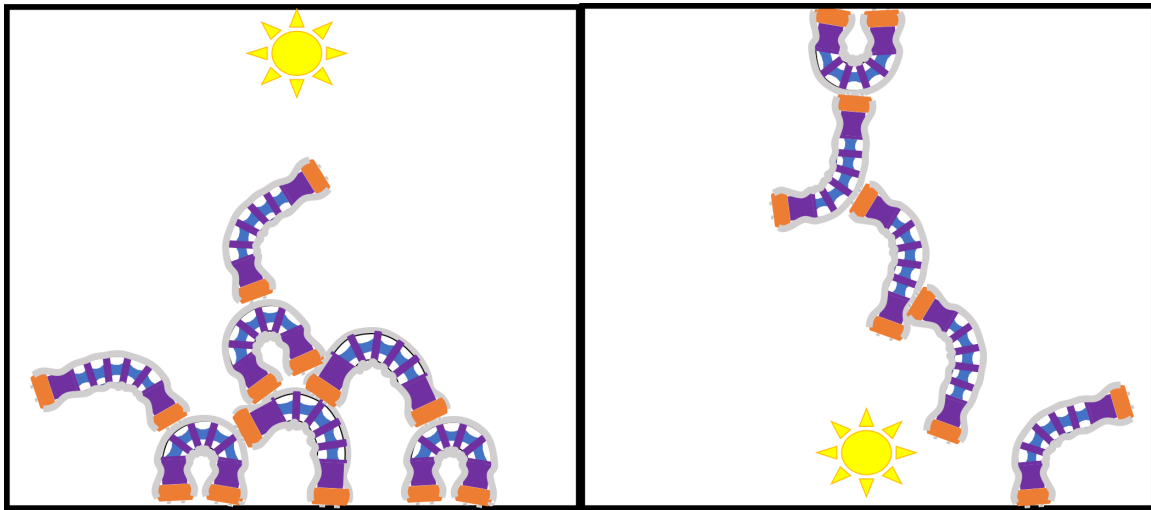
A singular robot that successfully completes a task 80% of the time may be relatively effective, especially for the purposes of research. The success rate drops significantly, however, with each added robot, especially if errors in a single robot are allowed to effect the behavior in the group. In the case of *E. robotica*, robots are contained in a single track, so it is likely that individual robot errors will effect the behavior of other robots in the system.

To scale the current system, priority should therefore be given to building error tolerance within the system by:

1. **Optimizing the control algorithm to prevent system errors** The first step is to eliminate system errors which occur when robots are behaving correctly. For *E. robotica*, from Chapter 5, this includes optimizing control parameters so that robots do not leave the structure early and can reattach if they sense another robot while in the flipping state.
2. **Preventing individual errors from becoming system failures** Currently if a robot runs out of battery or becomes stuck, it is usually necessary to stop the experiment or in some way rescue the robot. As we scale up, this becomes untenable. Robot failures must, if possible, not require rescue, and allow other robots to continue on the mission.
3. **Reducing overall fatal errors for individual robots.** As discussed in Chapter 5, we can improve overall performance of the robot and prevent errors from disabling the robot by:
  - Improving attachment by adding feedback and improved calibration of IR sensors.
  - Adding logic to allow the robot to sense if it is "stuck" (e.g. fully bent but without a new surface to attach to), and to either move backwards or simply freeze in place (so as not to impede other robots).
  - Increasing battery life. Each board currently runs off a 120 mAh battery, which lasts for about 20-30 minutes of testing when robots are fully actuated. A larger battery could be incorporated, possibly as a structural feature in order to save weight. A battery indicator could also prevent failures due to battery life. Since the PCB boards run on two separate batteries, this is especially important.

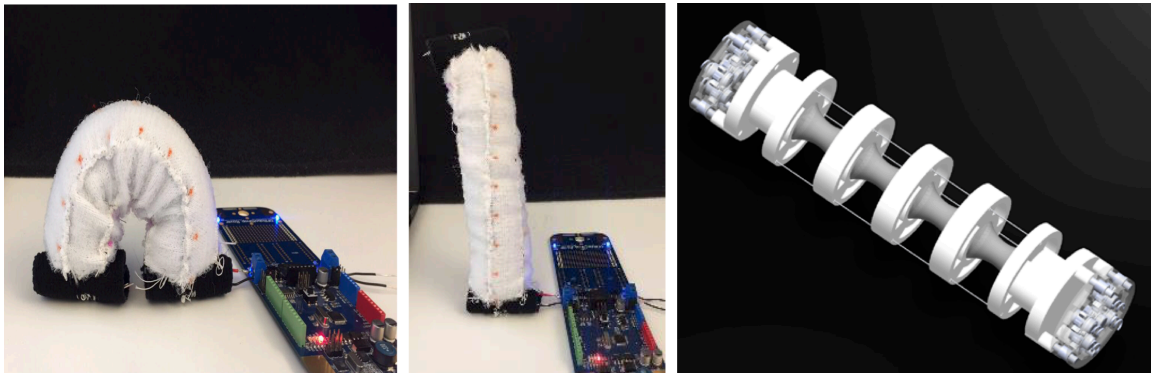
Lastly, robots in a larger swarm should be manufactured efficiently and in a consistent fashion. Improvements to manufacturing would make the *E. robotica* system faster and more homogeneous. Improvements that positively effect error tolerance of individual robots (see point 3) should be prioritized, specifically increasing the robustness of the tension switches as discussed in Section 3.3.2, improving precision of IR placement (Section 3.5.2) and Velcro attachment (Section 3.6.1), and developing an improved IR sensor calibration method (Section 3.5.2). Simple improvements to soldering methods, wire routing, etc, can also have a positive effect on the overall robustness of the robot modules.

With these improvements, I can scale up the *E. robotica* system, to form and dissolve structures with many robots and to try new terrains. By adding additional sensing, such as a light sensor, color sensor or camera on the gripper, the current robot may also be able to form goal driven structures such as towers or chains, as shown in Fig. 6.1. As discussed in Chapter 4, rules for these formations could be tested in the current simulation framework.



**Figure 6.1:** Tower and Chain formation using the same local rules and a set goal.





**Figure 6.2:** The Floppy robot, which is made of foam and covered completely in the stretch Velcro is also able to flip using cables. (Left) and 3D rendering of *E. robotica* with triangulated cables (Right)

### 6.2.2 MOVING TO 3D

The current robot, although able to climb over other robots in the vertical plane, is limited to that 2D plane and traveling in a single direction. Multi-directional traffic and 3D movement are not considered, although the natural system achieves both: While migration pathways of army ants will generally move in one direction, foraging paths contain both ants leaving the nest and returning from a raiding site. We believe the local control rules should extend to traffic from multiple directions and 3D movement; however, both require a 3D simulator and hardware platform to test, since agents traveling in multiple directions must be able to pass each other.

For practical applications, it will be essential to expand both the hardware and software may be expanded to 3D. Ideas for 3D robots are shown in Fig. 6.2. On the left is a softer robot, “Floppy,” where we replaced the flexible and stiff inner portion of the body with foam and covered it in stretchy velcro. This was inspired by the plush robots in Bern et al.<sup>98</sup>. The current robot may also be adapted to 3D; a rendering is shown on the right in Fig. 6.2. Both robots would use the same cable driven method to flip, but can control their direction of travel by replacing the two opposing cables with three triangulated cables, similar again to Treebot<sup>74</sup>.

Thus, flipping robot modules may be created for use in 3D adaptive self-assembly.

In order to create 3D structures, we must also create traffic build-up along a pathway. In our 2D set-up, both in simulation and in the hardware implementation, we were able to constrain the robots to a single path. Ants similarly maintain constrained pathways through their trails, but the path is not limited to a single line. They are allowed to stray from the path but encouraged to stay on the established trail by the pheromones laid down by their predecessors. Without these pheromone paths, ants might simply spread out, avoid each other, and never assemble bridges.

Similarly, to create 3D ant-like structures, we must allow robot agents space to wander, while still constraining them to a set path. We are thus posed with the question of *how to recreate ant-like pheromone trails in robots*. The problem is similar to many path planning in robotics, except that we wish to create rather than prevent robot collisions. We also want the system to remain de-centralized. For a 3D adaptive self-assembly system:

- **Robot agents must have a set goal(s).** Robots must not to walk randomly, but, like ants traveling to or from the nest, should have a set goal direction.
- **Robot agents must be able to sense the goal and the trail path.**

One way to do this would be to create a set goal position or direction and spawn robots with the same or similar starting position, much as we did for the simulation in Chapter 4. If robots are programmed to travel along the straightest path from the start position to the goal, collisions can occur. This is one of the simplest methods to try in both simulation and hardware and may be realized in hardware with directional sensors such as a magnetometer or by using a light source as a goal point.

A second, more complex method is to define the pheromone trail as a potential field. This could be achieved in hardware in a few ways. “Smart” environments such as the Kilogrid<sup>99</sup> could communicate values to the robot. Color sensors could also enable robots to sense a potential field either on such a smart environment or a colored surface. A smart environment might require a new gripper, but custom printed fabric could be used with an adapted version of the current robot such as the Floppy robot in Fig. 6.2.

These are just a few ideas and expansions to the work of this thesis. Since adaptive and soft systems in self-assembly are relatively new and unexplored, there is incredible potential and possibility for new designs, algorithms, and applications.

# Appendix A

## Simulation in Pymunk

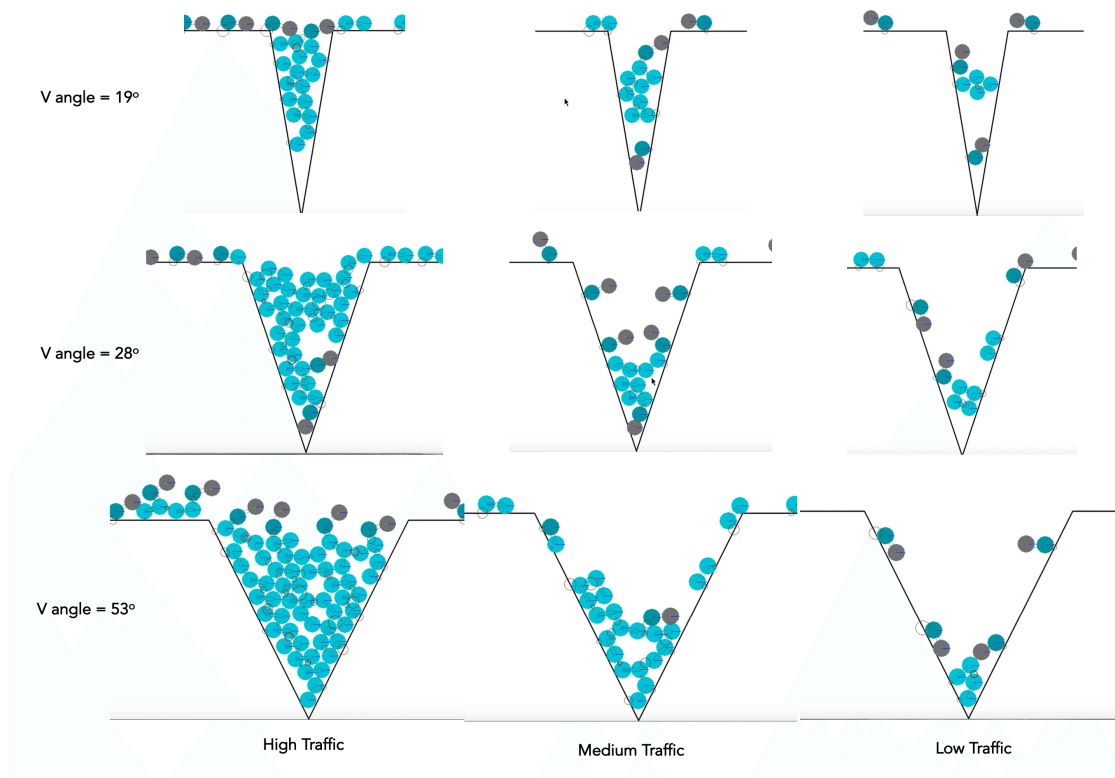
Before the final simulation experiments were performed in Box2D (Chapter 4), we first developed a simulation model in Pymunk, a python wrapper for the physics simulator, Chipmunk2. The simulation environment and model of the robot was developed by David Becerra, during a summer undergraduate research experience. By using and adapting the simulator software that he developed, I was able to form bridges that responded to traffic levels and size of the terrain. I ran the following experiments: 50 on V-bridge formation, 19 on downwards ramps, 13 on upwards ramps and 16 on formation and dissolution dissolution). I used these initial simulations to better understand the interaction between the various parameters involved bridge building, and the effect of different rules. Examples of bridge building from these initial simulations are

shown in shown in Fig. A.1.

The robot follows the same simple rule discussed in Chapter 4 - if it is gripped, it enters a bridge state, and stays in place. The robot model uses the same circular grippers connected with pin joints to the rectangular center. However, the robot agents have a much larger degree of compliance; pin joints in Pymunk are inherently compliant and, more problematically, users appear to have little access to control or tune this compliance. Some of this also appears to be stochastic which led to robots with varying step size, and different bridge formations, even for the same parameter choices.

The results in this section follow similar trends as the final simulations - bridge height increases with higher traffic levels and with narrower gaps. At very low traffic levels, in wide V-gaps, no bridge formed. In these initial simulations, at high traffic levels, the robots were more likely to completely fill the bridge, even when the bridge eventually stabilized. This could be due to a few differences between the Pymunk simulation and the implementation in Box2D. In her thesis, Houel discusses an additional rule comparison: in the results presented in Chapter 4, if two robots sense contact, both robots will join the bridge, even if only the bottom robot is gripped. In her additional testing, she tried the case where only the bottom robot joined the bridge and observed that this led to a filling of the V gap, but significantly reduced the success rate of dissolution to 33% total dissolution<sup>93</sup>. This is similar to the results that we saw in Pymunk, though my results for dissolution were lower - none of the bridges formed were able to dissolve. As discussed previously, the Pymunk simulated robot model has substantially more compliance, which could affect the formation of bridges.

One of the key lessons learned from these simulations was the need for the angle limit (robot goes through neutral) condition described in Chapter 4. This helps to prevent many of the up-



**Figure A.1:** Bridges formed in simulation in varying V-shaped terrains, with narrow Vs at the top, medium in the center and wide at the bottom. Traffic decreases from left to right.

grab and other errors which can prevent dissolution from occurring.

As in the simulations in Chapter 4, traffic levels were controlled via a spawn delay time. The distance between robots for this spawn delay was determined 8 traffic levels were tried, with the distance from robot to robot varying from 1.25 BL to 3.13 BL. In 4 of these 8 traffic conditions, robots were in sync with each other; the other four were chosen to be out of phase with each other.

I also tested ramp formation. While I collected fewer data points for these formations, they followed a similar trend as the V-shaped terrains, but for very wide Vs, since a ramp is essentially

a rotated V with a  $90^\circ$  angle.

We were not able to achieve bridge dissolution except in some rare cases, where the number of robots was kept less than ten. Even here, we achieved only partial dissolution, with a few robots getting stuck in double grabs, as discussed earlier in Chapter 4 Fig. 4.12.

## References

- [1] S. Powell and N. Franks, “How a few help all: living pothole plugs speed prey delivery in the army ant *Eciton burchellii*,” *Animal Behaviour*, vol. 73, pp. 1067–1076, 6 2007.
- [2] S. Garnier, T. Murphy, M. Lutz, E. Hurme, S. Leblanc, and I. D. Couzin, “Stability and responsiveness in a self-organized living architecture,” *PLoS computational biology*, vol. 9, p. e1002984, 1 2013.
- [3] C. R. Reid, M. J. Lutz, S. Powell, A. B. Kao, I. D. Couzin, and S. Garnier, “Army ants dynamically adjust living bridges in response to a cost–benefit trade-off,” *Proceedings of the National Academy of Sciences*, vol. 112, no. 49, p. 201512241, 2015.
- [4] C. Anderson, G. Theraulaz, and J.-L. Deneubourg, “Self-assemblages in insect societies,” *Insectes Sociaux*, vol. 49, pp. 99–110, 5 2002.
- [5] M. Brambilla, E. Ferrante, M. Birattari, and M. Dorigo, “Swarm robotics: a review from the swarm engineering perspective,” *Swarm Intelligence*, vol. 7, pp. 1–41, 3 2013.
- [6] M. Rubenstein, a. Cornejo, and R. Nagpal, “Programmable self-assembly in a thousand-robot swarm,” *Science*, vol. 345, pp. 795–799, 8 2014.
- [7] M. Gauci, R. Nagpal, and M. Rubenstein, “Programmable Self-disassembly for Shape Formation in Large-Scale Robot Collectives,” in *Distributed Autonomous Robotic Systems*, pp. 573–586, Springer, Cham, 2018.
- [8] I. Slavkov, D. Carrillo-Zapata, N. Carranza, X. Diego, F. Jansson, J. Kaandorp, S. Hauert, and J. Sharpe, “Morphogenesis in robot swarms,” *Science Robotics*, vol. 3, p. eaau9178, 12 2018.
- [9] J. Werfel, K. Petersen, and R. Nagpal, “Designing Collective Behavior in a Termite-Inspired Robot Construction Team,” *Science*, vol. 343, pp. 754–758, 2 2014.



- [10] A. Campo, S. Nouyan, M. Birattari, R. Groß, and M. Dorigo, “Negotiation of Goal Direction for Cooperative Transport,” in *International Workshop on Ant Colony Optimization and Swarm Intelligence*, pp. 191–202, Springer, Berlin, Heidelberg, 2006.
- [11] M. Rubenstein, A. Cabrera, J. Werfel, G. Habibi, J. McLurkin, and R. Nagpal, “Collective transport of complex objects by simple robots: Theory and experiments,” *12th International Conference on Autonomous Agents and Multiagent Systems 2013, AAMAS 2013*, vol. 1, pp. 47–54, 2013.
- [12] F. Mondada, L. L. M. Gambardella, D. Floreano, S. Nolfi, J. L. Deneubourg, and M. Dorigo, “The cooperation of swarm-bots: Physical interactions in collective robotics,” *IEEE Robotics and Automation Magazine*, vol. 12, pp. 21–28, 6 2005.
- [13] R. Groß, M. Bonani, F. Mondada, and M. Dorigo, “Autonomous self-assembly in a swarm-bot,” *Proceedings of the 3rd International Symposium on Autonomous Minirobots for Research and Edutainment, AMiRE 2005*, vol. 22, no. 6, pp. 314–322, 2006.
- [14] M. Shimizu and A. Ishiguro, “An amoeboid modular robot that exhibits real-time adaptive reconfiguration,” in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1496–1501, IEEE, 10 2009.
- [15] M. Yim, Y. Z. Y. Zhang, and D. Duff, “Modular robots,” *IEEE Spectrum*, 2002.
- [16] J. Paulos, N. Eckenstein, T. Tosun, J. Seo, J. Davey, J. Greco, V. Kumar, and M. Yim, “Automated Self-Assembly of Large Maritime Structures by a Team of Robotic Boats,” *IEEE Transactions on Automation Science and Engineering*, vol. 12, pp. 958–968, 7 2015.
- [17] J. W. Romanishin, J. Mamish, and D. Rus, “Decentralized Control for 3D M-Blocks for Path Following, Line Formation, and Light Gradient Aggregation,” in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4862–4868, IEEE, 11 2019.
- [18] M. Yim, W.-m. Shen, B. Salemi, D. Rus, M. Moll, H. Lipson, E. Klavins, and G. Chirikjian, “Modular Self-Reconfigurable Robot Systems [Grand Challenges of Robotics],” *IEEE Robotics & Automation Magazine*, vol. 14, pp. 43–52, 3 2007.
- [19] K. Gilpin, A. Knaian, and D. Rus, “Robot pebbles: One centimeter modules for programmable matter through self-disassembly,” in *2010 IEEE International Conference on Robotics and Automation*, pp. 2485–2492, IEEE, 5 2010.

- [20] E. Klavins, “Programmable Self-Assembly,” *IEEE Control Systems Magazine*, vol. 27, pp. 43–56, 8 2007.
- [21] B. Haghighat, E. Droz, and A. Martinoli, “Lily: A miniature floating robotic platform for programmable stochastic self-assembly,” in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1941–1948, IEEE, 2015.
- [22] C. Liu, M. Whitzer, and M. Yim, “A Distributed Reconfiguration Planning Algorithm for Modular Robots,” *IEEE Robotics and Automation Letters*, vol. 4, pp. 4231–4238, 10 2019.
- [23] N. Cheney, R. MacCurdy, J. Clune, and H. Lipson, “Unshackling evolution,” *ACM SIGEVOlution*, vol. 7, pp. 11–23, 8 2014.
- [24] A. Vergara, Y.-s. Lau, R.-F. Mendoza-Garcia, and J. C. Zagal, “Soft Modular Robotic Cubes: Toward Replicating Morphogenetic Movements of the Embryo,” *PLOS ONE*, vol. 12, p. e0169179, 1 2017.
- [25] M. T. Tolley, R. F. Shepherd, B. Mosadegh, K. C. Galloway, M. Wehner, M. Karpelson, R. J. Wood, and G. M. Whitesides, “A Resilient, Untethered Soft Robot,” *Soft Robotics*, vol. 1, pp. 213–223, 9 2014.
- [26] M. Malley, M. Rubenstein, and R. Nagpal, “Flippy: A soft, autonomous climber with simple sensing and control,” in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6533–6540, IEEE, 9 2017.
- [27] P. S. M. Hill, “Vibration and Animal Communication: A Review,” *American Zoologist*, vol. 41, pp. 1135–1142, 10 2001.
- [28] N. J. Mlot, C. A. Tovey, and D. L. Hu, “Fire ants self-assemble into waterproof rafts to survive floods,” *Proceedings of the National Academy of Sciences of the United States of America*, vol. 108, pp. 7669–73, 5 2011.
- [29] G. M. Whitesides, “Self-Assembly at All Scales,” *Science*, vol. 295, no. 5564, pp. 2418–2421, 2002.
- [30] J. Wojtusiak, E. J. Godzińska, and A. Dejean, “Capture and retrieval of very large prey by workers of the African weaver ant, *Oecophylla longinoda* (Latreille 1802),” *Tropical Zoology*, vol. 8, pp. 309–318, 11 1995.
- [31] B. K. Hölldobler and E. O. Wilson, “Weaver Ants,” *Scientific American*, vol. 237, no. 6, pp. 146–154, 1977.

- [32] M. Jorgensen, E. Ostergaard, and H. Lund, "Modular ATRON: modules for a self-reconfigurable robot," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No. 04CH37566)*, vol. 2, pp. 2068–2073, IEEE, 2004.
- [33] H. Kurokawa, K. Tomita, A. Kamimura, S. Kokaji, T. Hasuo, and S. Murata, "Distributed Self-Reconfiguration of M-TRAN III Modular Robotic System," *The International Journal of Robotics Research*, vol. 27, pp. 373–386, 3 2008.
- [34] K. Gilpin, K. Koyanagi, and D. Rus, "Making self-disassembling objects with multiple components in the Robot Pebbles system," in *2011 IEEE International Conference on Robotics and Automation*, pp. 3614–3621, IEEE, 5 2011.
- [35] J. W. Romanishin, K. Gilpin, S. Claici, and D. Rus, "3D M-Blocks: Self-reconfiguring robots capable of locomotion via pivoting in three dimensions," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1925–1932, IEEE, 5 2015.
- [36] L. Cucu, M. Rubenstein, and R. Nagpal, "Towards self-assembled structures with mobile climbing robots," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1955–1961, IEEE, 5 2015.
- [37] Byoung Kwon An, "Em-cube: cube-shaped, self-reconfigurable robots sliding on structure surfaces," in *2008 IEEE International Conference on Robotics and Automation*, pp. 3149–3155, IEEE, 5 2008.
- [38] A. Castano, A. Behar, and P. Will, "The Conro modules for reconfigurable robots," *IEEE/ASME Transactions on Mechatronics*, vol. 7, pp. 403–409, 12 2002.
- [39] J. Davey, N. Kwok, and M. Yim, "Emulating self-reconfigurable robots - design of the SMORES system," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4464–4469, IEEE, 10 2012.
- [40] K. Hosokawa, T. Tsujimori, T. Fujii, H. Kaetsu, H. Asama, Y. Kuroda, and I. Endo, "Self-organizing collective robots with morphogenesis in a vertical plane," in *Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No. 98CH36146)*, vol. 4, pp. 2858–2863, IEEE, 1998.
- [41] H. Kurokawa, S. Murata, E. Yoshida, K. Tomita, and S. Kokaji, "A 3-D self-reconfigurable structure and experiments," in *Proceedings. 1998 IEEE/RSJ International Conference on Intelligent Robots and Systems. Innovations in Theory, Practice and Applications (Cat. No. 98CH36190)*, vol. 2, pp. 860–865, IEEE, 1998.

- [42] S. Murata, E. Yoshida, A. Kamimura, H. Kurokawa, K. Tomita, and S. Kokaji, "M-TRAN: self-reconfigurable modular robotic system," *IEEE/ASME Transactions on Mechatronics*, vol. 7, pp. 431–441, 12 2002.
- [43] C. H. R. M. & S. B. Shen, W. M., "Rolling and climbing by the multifunctional superbot reconfigurable robotic system," in *Proceedings of the Space Technology International Forum*, pp. 839–848, 2008.
- [44] K. Kotay and D. Rus, "Locomotion versatility through self-reconfiguration," *Robotics and Autonomous Systems*, vol. 26, pp. 217–232, 2 1999.
- [45] I. O'Hara, J. Paulos, J. Davey, N. Eckenstein, N. Doshi, T. Tosun, J. Greco, J. Seo, M. Turpin, V. Kumar, and M. Yim, "Self-assembly of a swarm of autonomous boats into floating structures," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1234–1240, IEEE, 5 2014.
- [46] P. Swisler and M. Rubenstein, "FireAnt: A Modular Robot with Full-Body Continuous Docks," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 6812–6817, 2018.
- [47] Z. Chen, C. Zhao, Y. Zhang, Y. Zhu, J. Fan, and J. Zhao, "C-Balls: A Modular Soft Robot Connected and Driven via Magnet Forced," *Journal of Physics: Conference Series*, vol. 1207, p. 012006, 4 2019.
- [48] B. Chu, K. Jung, C.-S. Han, and D. Hong, "A survey of climbing robots: Locomotion and adhesion," *International Journal of Precision Engineering and Manufacturing*, vol. 11, pp. 633–647, 8 2010.
- [49] R. Siegwart, P. Lamon, T. Estier, M. Lauria, and R. Piguet, "Innovative design for wheeled locomotion in rough terrain," *Robotics and Autonomous Systems*, vol. 40, pp. 151–162, 8 2002.
- [50] N. Rahman, "MOBIT, A Small Wheel - Track - Leg Mobile Robot," in *2006 6th World Congress on Intelligent Control and Automation*, vol. 2, pp. 9159–9163, IEEE, 2006.
- [51] T. Allen, R. Quinn, R. Bachmann, and R. Ritzmann, "Abstracted biological principles applied with reduced actuation improve mobility of legged vehicles," in *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No.03CH37453)*, vol. 2, pp. 1370–1375, IEEE, 2003.

- [52] B. Lambrecht, A. Horchler, and R. Quinn, "A Small, Insect-Inspired Robot that Runs and Jumps," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pp. 1240–1245, IEEE, 2005.
- [53] A. M. Johnson, M. T. Hale, G. C. Haynes, and D. E. Koditschek, "Autonomous legged hill and stairwell ascent," *9th IEEE International Symposium on Safety, Security, and Rescue Robotics, SSRR 2011*, pp. 134–142, 2011.
- [54] U. Saranli, "RHex: A Simple and Highly Mobile Hexapod Robot," *The International Journal of Robotics Research*, vol. 20, pp. 616–631, 7 2001.
- [55] B. He, Z. Wang, M. Li, K. Wang, R. Shen, and S. Hu, "Wet Adhesion Inspired Bionic Climbing Robot," *IEEE/ASME Transactions on Mechatronics*, vol. 19, pp. 312–320, 2014.
- [56] R. Chen, R. Liu, and H. Shen, "Design of a double-tracked wall climbing robot based on electrostatic adhesion mechanism," in *2013 IEEE Workshop on Advanced Robotics and its Social Impacts*, pp. 212–217, IEEE, 11 2013.
- [57] H. Prahlad, R. Pelrine, S. Stanford, J. Marlow, and R. Kornbluh, "Electroadhesive robots—wall climbing robots enabled by a novel, robust, and electrically controllable adhesion technology," in *2008 IEEE International Conference on Robotics and Automation*, pp. 3028–3033, IEEE, 5 2008.
- [58] B. Kennedy, A. Okon, H. Aghazarian, M. Badescu, X. Bao, Y. Bar-Cohen, Z. Chang, B. E. Dabiri, M. Garrett, L. Magnone, and S. Sherrit, "Lemur IIb: a robotic system for steep terrain access," *Industrial Robot: An International Journal*, vol. 33, pp. 265–269, 7 2006.
- [59] E. W. Hawkes, D. L. Christensen, and M. R. Cutkosky, "Vertical dry adhesive climbing with a 100× bodyweight payload," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3762–3769, IEEE, 5 2015.
- [60] D. Santos, B. Heyneman, S. Kim, N. Esparza, and M. R. Cutkosky, "Gecko-inspired climbing behaviors on vertical and overhanging surfaces," in *2008 IEEE International Conference on Robotics and Automation*, pp. 1125–1131, IEEE, 5 2008.
- [61] A. Asbeck, M. Cutkosky, and W. Provancher, "SpinybotII: climbing hard walls with compliant microspines," *ICAR '05. Proceedings., 12th International Conference on Advanced Robotics, 2005.*, pp. 601–606, 2005.

- [62] Y. Liu, S. Sun, X. Wu, and T. Mei, "A Wheeled Wall-Climbing Robot with Bio-Inspired Spine Mechanisms," *Journal of Bionic Engineering*, vol. 12, pp. 17–28, 1 2015.
- [63] M. P. Murphy, C. Kute, Y. Mengüç, and M. Sitti, "Waalbot II: Adhesion Recovery and Improved Performance of a Climbing Robot using Fibrillar Adhesives," *The International Journal of Robotics Research*, vol. 30, pp. 118–133, 1 2011.
- [64] P. Schoeneich, F. Rochat, O. T.-D. Nguyen, R. Moser, and F. Mondada, "TRIPILLAR: a miniature magnetic caterpillar climbing robot with plane transition ability," *Robotica*, vol. 29, pp. 1075–1081, 4 2011.
- [65] T. Seo and M. Sitti, "Tank-Like Module-Based Climbing Robot Using Passive Compliant Joints," *IEEE/ASME Transactions on Mechatronics*, vol. 18, pp. 397–408, 2 2013.
- [66] R. L. Tummala, R. Mukherjee, N. Xi, D. Aslam, H. Dulimarta, J. Xiao, M. Minor, and G. Dang, "Climbing the Walls," *IEEE Robotics & Automation Magazine*, vol. 9, no. 4, pp. 10–19, 2002.
- [67] Y. Guan, H. Zhu, W. Wu, X. Zhou, L. Jiang, C. Cai, L. Zhang, and H. Zhang, "A Modular Biped Wall-Climbing Robot With High Mobility and Manipulating Function," *IEEE/ASME Transactions on Mechatronics*, vol. 18, pp. 1787–1798, 12 2013.
- [68] Y. Yoshida and S. Ma, "Design of a wall-climbing robot with passive suction cups," in *2010 IEEE International Conference on Robotics and Biomimetics*, pp. 1513–1518, IEEE, 12 2010.
- [69] J. Shang, T. Sattar, S. Chen, and B. Bridge, "Design of a climbing robot for inspecting aircraft wings and fuselage," *Industrial Robot*, vol. 34, no. 6, pp. 495–502, 2007.
- [70] K. a. Daltorio, T. E. Wei, a. D. Horschler, L. Southard, G. D. Wile, R. D. Quinn, S. N. Gorb, and R. E. Ritzmann, "Mini-Whegs TM Climbs Steep Surfaces Using Insect-inspired Attachment Mechanisms," *The International Journal of Robotics Research*, vol. 28, pp. 285–302, 2 2009.
- [71] M. J. Spenko, G. C. Haynes, J. A. Saunders, M. R. Cutkosky, A. A. Rizzi, R. J. Full, and D. E. Koditschek, "Biologically inspired climbing with a hexapedal robot," *Journal of Field Robotics*, vol. 25, pp. 223–242, 4 2008.
- [72] S. Kim, M. Spenko, S. Trujillo, B. Heyneman, V. Mattoli, and M. R. Cutkosky, "Whole body adhesion: hierarchical, directional and distributed control of adhesive forces for a climbing robot," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pp. 1268–1273, IEEE, 4 2007.

- [73] Y. Liu, X. Wu, H. Qian, D. Zheng, J. Sun, and Y. Xu, "System and design of Cloth-bot: A robot for flexible clothes climbing," in *2012 IEEE International Conference on Robotics and Automation*, pp. 1200–1205, IEEE, 5 2012.
- [74] T. L. Lam and Y. Xu, "A flexible tree climbing robot: Treebot - design and implementation," in *2011 IEEE International Conference on Robotics and Automation*, pp. 5849–5854, IEEE, 5 2011.
- [75] M. S. Verma, A. Ainla, D. Yang, D. Harburg, and G. M. Whitesides, "A Soft Tube-Climbing Robot," *Soft Robotics*, vol. 5, no. 2, pp. 133–137, 2018.
- [76] G. Gu, J. Zou, R. Zhao, X. Zhao, and X. Zhu, "Soft wall-climbing robots," *Science Robotics*, vol. 3, p. eaat2874, 12 2018.
- [77] A. Sirken, G. Knizhnik, J. McWilliams, and S. Bergbreiter, "Bridge risk investigation diagnostic grouped exploratory (BRIDGE) bot," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6526–6532, IEEE, 9 2017.
- [78] A. Asbeck, S. Dastoor, A. Parness, L. Fullerton, N. Esparza, D. Soto, B. Heyneman, and M. Cutkosky, "Climbing rough vertical surfaces with hierarchical directional adhesion," in *2009 IEEE International Conference on Robotics and Automation*, pp. 2675–2680, IEEE, 5 2009.
- [79] J. Xu, L. Xu, J. Liu, X. Li, and X. Wu, "Survey on Bioinspired Adhesive Methods and Design and Implementation of A Multi-Mode Biomimetic Wall-Climbing Robot," in *2018 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, pp. 688–693, IEEE, 7 2018.
- [80] H.-T. Lin, G. G. Leisk, and B. Trimmer, "GoQBot: a caterpillar-inspired soft-bodied rolling robot," *Bioinspiration & Biomimetics*, vol. 6, p. 026007, 6 2011.
- [81] R. Full, K. Earls, M. Wong, and R. Caldwell, "Locomotion like a wheel?," *Nature*, vol. 365, pp. 495–495, 10 1993.
- [82] A. Dollar and R. Howe, "A robust compliant grasper via shape deposition manufacturing," *IEEE/ASME Transactions on Mechatronics*, vol. 11, pp. 154–161, 4 2006.
- [83] J. Gafford, Y. Ding, A. Harris, T. McKenna, P. Polygerinos, D. Holland, A. Moser, and C. Walsh, "Shape Deposition Manufacturing of a Soft, Atraumatic, Deployable Surgical Grasper," *Journal of Medical Devices*, vol. 8, p. 030927, 7 2014.

- [84] M. Wu, G. Pan, T. Zhang, S. Chen, F. Zhuang, and Z. Yan-zheng, “Design and Optimal Research of a Non-Contact Adjustable Magnetic Adhesion Mechanism for a Wall-Climbing Welding Robot,” *International Journal of Advanced Robotic Systems*, vol. 10, p. 63, 1 2013.
- [85] A. Parness, M. Frost, N. Thatte, J. P. King, K. Witkoe, M. Nevarez, M. Garrett, H. Agazarian, and B. Kennedy, “Gravity-independent Rock-climbing Robot and a Sample Acquisition Tool with Microspine Grippers,” *Journal of Field Robotics*, vol. 30, pp. 897–915, 11 2013.
- [86] A. Dejean, C. Leroy, B. Corbara, O. Roux, R. Céréghino, J. Orivel, and R. Boulay, “Arboreal ants use the “Velcro® principle” to capture very large prey,” *PloS one*, vol. 5, p. e11331, 1 2010.
- [87] G. Espinosa and M. Rubenstein, “Using Hardware Specialization and Hierarchy to Simplify Robotic Swarms,” in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 7667–7673, IEEE, 5 2018.
- [88] H. Wang and M. Rubenstein, “Shape Formation in Homogeneous Swarms Using Local Task Swapping,” *IEEE Transactions on Robotics*, pp. 1–16, 2020.
- [89] G. Jing, T. Tosun, M. Yim, and H. Kress-Gazit, “An end-to-end system for accomplishing tasks with modular robots: Perspectives for the AI community,” *IJCAI International Joint Conference on Artificial Intelligence*, pp. 4879–4883, 2017.
- [90] J. Daudelin, G. Jing, T. Tosun, M. Yim, H. Kress-Gazit, and M. Campbell, “An integrated system for perception-driven autonomy with modular robots,” *Science Robotics*, vol. 3, p. eaat4983, 10 2018.
- [91] N. Melenbrink, P. Michalatos, P. Kassabian, and J. Werfel, “Using local force measurements to guide construction by distributed climbing robots,” in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4333–4340, IEEE, 9 2017.
- [92] R. O’Grady, R. Groß, F. Mondada, M. Bonani, and M. Dorigo, “Self-assembly on demand in a group of physical autonomous mobile robots navigating rough terrain,” *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 3630 LNAI, pp. 272–281, 2005.
- [93] L. Houel, “Self-assembly of soft-robots in simulation inspired by army ant bridge behavior,” Master’s thesis, École polytechnique fédérale de Lausanne, 2019.



- [94] S. Fuchs, “The response to vibrations of the substrate and reactions to the specific drumming in colonies of carpenter ants (*Camponotus*, Formicidae, Hymenoptera),” *Behavioral Ecology and Sociobiology*, vol. 1, no. 2, pp. 155–184, 1976.
- [95] B. Hölldobler, “Multimodal signals in ant communication,” *Journal of Comparative Physiology A: Sensory, Neural, and Behavioral Physiology*, vol. 184, pp. 129–141, 3 1999.
- [96] C. C. Vu and J. Kim, “Simultaneous Sensing of Touch and Pressure by Using Highly Elastic e-Fabrics,” *Applied Sciences*, vol. 10, p. 989, 2 2020.
- [97] O. Atalay, A. Atalay, J. Gafford, and C. Walsh, “A Highly Sensitive Capacitive-Based Soft Pressure Sensor Based on a Conductive Fabric and a Microporous Dielectric Layer,” *Advanced Materials Technologies*, vol. 3, p. 1700237, 1 2018.
- [98] J. M. Bern, G. Kumagai, and S. Coros, “Fabrication, modeling, and control of plush robots,” in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3739–3746, IEEE, 9 2017.
- [99] G. Valentini, A. Antoun, M. Trabatttoni, B. Wiandt, Y. Tamura, E. Hocquard, V. Trianni, and M. Dorigo, “Kilogrid: a novel experimental environment for the Kilobot robot,” *Swarm Intelligence*, vol. 12, pp. 245–266, 9 2018.