



THE UNIVERSITY *of* EDINBURGH

This thesis has been submitted in fulfilment of the requirements for a postgraduate degree (e.g. PhD, MPhil, DClinPsychol) at the University of Edinburgh. Please note the following terms and conditions of use:

This work is protected by copyright and other intellectual property rights, which are retained by the thesis author, unless otherwise stated.

A copy can be downloaded for personal non-commercial research or study, without prior permission or charge.

This thesis cannot be reproduced or quoted extensively from without first obtaining permission in writing from the author.

The content must not be changed in any way or sold commercially in any format or medium without the formal permission of the author.

When referring to this work, full bibliographic details including the author, title, awarding institution and date of the thesis must be given.



Modular Robots for Sorting

Ross McKenzie

Supervisors: Dr. Adam Stokes and Prof. Barbara Webb

A Thesis Submitted for the Degree of
Doctor of Philosophy
2019

Abstract

Current industrial sorting systems allow for low error, high throughput sorts with tightly constrained properties. These sorters, however, are often hardware limited to certain items and criteria. There is a need for more adaptive sorting systems for processes that involve a high throughput of heterogeneous items such as import testing by port authorities, warehouse sorting for online retailers, and sorting recycling. The variety of goods that need to be sorted in these applications mean that existing sorting systems are unsuitable, and the objects often need to be sorted by hand. In this work I detail my design and control of a modular, robotic sorting system using linear actuating robots to create both low-frequency vibrations and peristaltic waves for sorting. The adaptability of the system allows for multimodal sorting and can improve heterogeneous sorting systems.

I have designed a novel modular robot called the Linbot. These Linbots are based on an actuator design utilising a voice coil for linear motion. I designed this actuator to be part of a modular robot by adding on-board computation, sensing and communication. I demonstrate the individual characteristics of these robots through a series of experiments in order to give a comprehensive overview of their abilities. By combining multiple Linbots in a collective I demonstrate their ability to move and sort objects using cooperative peristaltic motion.

In order to allow for rapid optimisation of control schemes for Linbot collectives I required a peristaltic table simulator. I designed and implemented a peristaltic table simulator, called PeriSim, due to a lack of existing solutions. Controllers optimised in simulation often suffer from a reduction in performance when moved to a real-world system due to the inaccuracies in the simulation, this effect is called the reality gap. I used a method for reducing the reality gap called the *radical envelope of noise hypothesis*, whereby I only modelled the key aspects of peristalsis in PeriSim and then varied the underlying physics of the simulation between runs. I used PeriSim to optimise controllers in simulation that worked on a real-world system.

I demonstrate the how the Linbots and PeriSim can be used to build and control an adaptive sorter. I built an adaptive sorter made of a 5x5 grid of Linbots with a soft sheet covering them. I demonstrate that the sorter can grade produce and move objects of varying shapes and sizes. My work can guide the future design of industrial adaptive sorting systems.

Acknowledgements

Mohammed E. Sayed for help with the design, fabrication and testing of the Linbots. Jamie O. Roberts for assistance with the experimentation and data analysis. Brian Flynn for the design of the LC oscillator. Prof. Barbara Webb for her guidance on the structure and aims of the project. Dr. Adam Stokes for support with all aspects of this work. My work was supported by EPSRC via the Robotarium Capital Equipment and CDT Capital Equipment Grants and the University of Edinburgh and Heriot-Watt University CDT in Robotics and Autonomous Systems.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified. I have the copyright for all of the included publications.

A handwritten signature in black ink, appearing to read 'Ross McKenzie', written in a cursive style.

Ross McKenzie 15/02/2020

Contents

List of Figures	i
List of Tables	ii
Acronyms	iii
1 Introduction	1
1.1 Sorting	1
1.1.1 Multiple Sorting Processes Involve Multiple Types of Objects and Criteria	2
1.1.2 Open Loop Sorting Provides High Throughput for Certain Sort Criteria	2
1.1.3 Closed Loop Sorting Allows for a Larger Range of Criteria	3
1.1.4 Pick and Place Arms are Less Cost Efficient	3
1.1.5 Peristaltic Tables Allow for Robotic Sorting on a 2D Plane	4
1.2 Problem Statement and Breakdown	5
1.3 Preview of Contributions	6
1.4 Publications	7
1.4.1 Directly Related to this Project	7
1.4.2 Other Contributions	7
1.5 Organisation of the Thesis	8
1.6 QR Codes	9
2 Literature Review	10
2.0.1 Building Peristaltic Tables	10
2.1 Sorting Algorithms for Swarm Robots	12
2.1.1 Spatial Clusters of Swarm Robots	12
2.1.2 Swarm Robots can Sort into Defined Areas	12
2.1.3 Swarm Robotic Collaboration to Move Objects	12
2.2 Architectures for Modular Robots	13
2.2.1 Modular Robots Share Characteristics with Swarm Robots	13
2.2.2 Rotational Joints Allows for Modular Structures	14
2.2.3 Robots can Move Around their Collectives	14
2.2.4 Expanding Modules can Perform Tasks in a Collective	14
2.2.5 V-SPA Modules use a Vacuum Source to Bend and Actuate Linearly	15
2.2.6 Voice Coils can Be Used for Linear Actuating Modular Robots	15
2.3 Learning System Dynamics for Collective Control	16
2.3.1 Evolutionary Algorithms can Optimise a Controller Without Prior Knowledge of the System	16

2.3.2	Reinforcement Learning Can Be Used with Modular Robots to Learn Gait	17
2.3.3	Distributed, Scalable Learning	18
2.3.4	State Machine Genetic Algorithms are Suitable For Discrete Environments	18
2.3.5	Evolved Behaviour Trees can Create Human Readable Controllers	18
2.3.6	Distributed Hormone Control Enables Robust Communication . .	19
2.3.7	Peristaltic Control Using a Probabilistic Automaton	19
2.4	Overcoming The Reality Gap	19
2.4.1	Human Editing to Adapt a Controller to the Real World	19
2.4.2	Real World Validation Steps can be Used to Improve Simulations	20
2.4.3	Adding Noise to Simulation Parameters Creates Robust Controllers	20
3	Design and Fabrication of the Linbot	22
3.1	Introduction	22
3.1.1	Mechanical Design	22
3.1.2	Electronic Design	23
3.1.3	Completed Linbots	23
3.2	Linbot Publication	23
3.3	Supplemental Videos	35
3.4	Conclusion	35
4	Simulating Peristaltic Tables and Optimising Peristaltic Waves	36
4.1	Introduction	36
4.1.1	Making a New Simulator	36
4.1.2	PyTorch	36
4.1.3	PeriSim	37
4.2	PeriSim Publication	37
4.3	Supplemental Videos	47
4.4	Conclusion	47
5	A Modular Robotic Adaptive Sorting System	48
5.1	Introduction	48
5.1.1	Design and Fabrication	48
5.1.2	Controlling the Sorter	48
5.1.3	An Adaptive Sorter Made from Linbots	49
5.2	Modular Robotic Sorting System Publication	49
5.3	Supplemental Videos	75
5.4	Conclusion	75
6	Discussion and future research	76
6.1	Future Developments	76
6.1.1	Integration with Additional Sensors	76
6.1.2	Advanced Vibrational Sorting	76
6.1.3	Integration of Central Advisor Systems	77
6.1.4	Online Learning	77
6.1.5	3D Collectives	77
6.1.6	Mixed Stream Sorting Application	77
6.2	Conclusion	78

References	80
Appendices	85
A 2D Designs	86
B Bill of Materials	87
C Electronic Design	88
D Designing the Linbots	90
D.1 Mechanical Design	90
D.2 Integrated Magnetic Communication	91
D.3 Sensing	92
D.4 PCB Design and Fabrication	92
E Fabrication	94
F Supplemental Information of Publications	96
F.1 Linbots: Soft Modular Robots Utilizing Voice Coils Supplimentary Data	96
G PeriSim README	103

List of Figures

1.1	A diagram showing a sorting process. (A) The initial system. (B) The result of a categorisation by colour. (C)The result of an ordering by size.	1
1.2	A diagram of a peristaltic table. (A) A flexible top layer. (B) Linear actuators. (C) Rigid base. (D) The object being sorted moves down the gradient created by the table. (E)-(F) Additional sensors and/or manipulators. Reproduced from M. Stommel and W. Xu [14] ©2015 IEEE. . . .	5
2.1	Process of object movement with inflating air chambers at different times. Based on caterpillar locomotion. Reproduced from Z. Deng et al. [20] ©2016 IEEE.	11
2.2	Spatial sorting swarm robots a) Initial setup b) Expected result. Reproduced from H. Ding and H. Hamann et al. [23] ©2014 Springer International Publishing Switzerland.	13
2.3	(A) Collectives of chain-like modular robots using YaMoRs. Reproduced from R. Moeckel et al. [27] ©2006 Emerald Publishing Limited. (B) A pair of modular robots with rotating halves using Molecubes. Reproduced from V. Zykov et al. [28] ©2007 IEEE.	14
2.4	An overview of a V-SPA module. Reproduced from M. A. Robertson and J. Paik [32] ©2017 The Authors, some rights reserved; exclusive licensee American Association for the Advancement of Science.	16
2.5	Overview of the Wormbot. (a) the voice coil actuators (b) the assembled Wormbot. Reproduced from M. P. Nemitz et al. [33] ©2016 The Authors; Mary Ann Liebert, Inc.	17
A.1	2D designs for Linbot components	86
B.1	Bill of Materials. Components cost £21.88 for each Linbot when buying enough for 10 Linbots	87
C.1	Linbot board. Component list in Figure B.1	88
C.2	Linbot schematic	89

List of Tables

1.1	A comparison of three sorting methods. Throughput is given for both a 25% and 50% removal of objects from main line rate.	4
-----	---	---

Acronyms

ANN Artificial Neural Network. 20

EA Evolutionary Algorithm. 16–18, 20

EBTs Evolved Behaviour Trees. 18–20

FEA Finite Element Analysis. 36

GA Genetic Algorithm. 17–19

I²C Inter-Integrated Circuit. 76, 92

LC Inductor-Capacitor. 23, 91

PCB Printed Circuit Board. 22, 23, 91–95

RL Reinforcement Learning. 17, 18

ROS The Robot Operating System. 7, 8

SCARA Selective Compliance Articulated Robot Arm. 3

V-SPA Vacuum-powered Soft Pneumatic Actuator. i, 15, 16

Chapter 1

Introduction

1.1 Sorting

Sorting is the act of placing objects in certain positions, which can be virtual or physical, based on certain criteria. Sorting objects can involve classification, where objects are moved to the same location as other objects fulfilling certain criteria. Sorting can also involve ordering, where objects are moved into individual locations based on their rank in certain criteria. These differences are demonstrated by Figure 1.1.

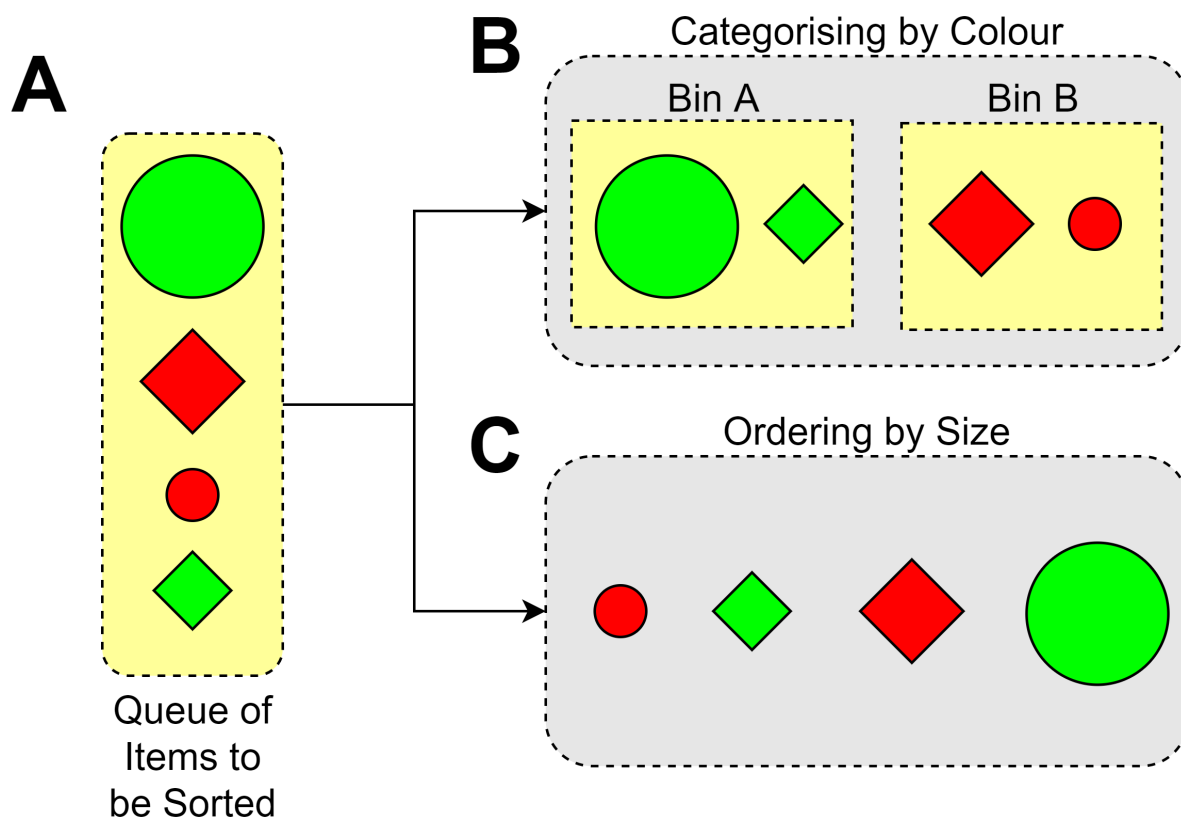


Figure 1.1: A diagram showing a sorting process. (A) The initial system. (B) The result of a categorisation by colour. (C) The result of an ordering by size.

Sorting is highly important in industry due to the need to sort products by type or remove objects that are unsuitable for purpose. Much industrial sorting involves highly

controlled inputs and only requires the sorting of one type of object. However, there are numerous processes that require the sorting of objects of various types via multiple criteria.

1.1.1 Multiple Sorting Processes Involve Multiple Types of Objects and Criteria

Recycling outputs must be of a high purity in order to be used in new products. The input to recycling plants is usually a diverse mixture of materials making the extraction of high purity materials demanding. In addition the differences in the rigidity, size and shapes of the items in the input stream mean that some are unsuitable for passing through certain sorting systems. Each different material can only be sorted by some systems leading to a web of interdependent sorting systems [1]. This problem is exacerbated by the use of mixed recycling bins, as in the UK, and the small amount of non-recyclable waste mixed into the input. Additionally the content of the input stream can change over time making existing systems sub-optimal. Pellegrinelli [2] writes that these changes cannot be adapted to by current automated sorting systems and so most plants rely on some amount of hand sorting. The author then suggests a theoretical macro-sorting system based on pick and place robots.

The shipping of groceries or other e-commerce products also requires the sorting of very different objects with little room for error. Zeng et al. [3] developed a pick and place system for moving items of varied shapes and sizes between bins using a robot arm guided by computer vision. Companies are also developing warehouse fulfilment systems using multiple robots that can pick and place containers [4]. Liang et al. [5] combine the two approaches with a robotic arm mounted on a mobile base capable of grasping and carrying objects.

Agricultural products can require separation from foreign matter after harvest and before any grading [6] requiring multiple sorting stages and machines. A single machine capable of both, removing foreign material and then grading items, could provide a more cost effective alternative.

Most food imports across the globe need to be correctly graded in order to comply with local import rules, for example non-EU rice imports to the UK [7]. To ensure compliance port authorities must conduct random or targeted inspections and select samples for laboratory testing. The sample sizes of these tests could be increased and testing time decreased given a system flexible enough to sort and grade a variety of objects. This would decrease port authority overheads.

1.1.2 Open Loop Sorting Provides High Throughput for Certain Sort Criteria

Mechanical sorting classifies or orders items through physical interaction in an open loop manner [8]. Items can be sieved to sort for size. In order to sort across the multiple dimensions of the item cylindrical rotating sieves are used which roll the items. Similar sorting can also be achieved by passing objects through a gap between

a roller and a wall, or another roller. Mechanical sorting by density can be achieved by placing items in a liquid of desired density and collecting those that float separately to those that sink. Vibrational sorting uses low-frequency oscillations of a surface to sort objects of different weights, densities and rigidity. This is achieved through the difference in velocities applied to the objects on top of the table. Most vibrational sorters use an inclined plane; as the plane vibrates objects are biased towards moving down the gradient. The size of this bias is determined by the physical characteristics of the objects on the plane. For example, less rigid objects can absorb more of the energy in each vibrational kick leading to lower velocity down the slope of the plane. Objects with smaller or smoother bases will slide further along the plane after each vibrational kick leading to greater velocity along the slope of the plane. This difference in velocity down a gradient can be used to distribute objects into different areas.

Mechanical sorters can be very high throughput and have relatively low costs. They are, however, highly inflexible and will have a very low range of sorting capabilities.

1.1.3 Closed Loop Sorting Allows for a Larger Range of Criteria

Robotic sorting relies on systems with sensors, computation and actuators. The sensors used are determined by the criteria the machine is sorting for. Robotic sorters can sort via the weight of objects. Optical sorting machines can use a camera or lasers for sensor data to sort based on a number of criteria, for example: size, shape and/or colour. Improvements to machine vision are facilitating better optical sorting machines. Image segmentation allows multiple objects at unknown positions to be passed through the sensor and machine learning classification is expanding the range of criteria used for sorting. Robotic sorters require actuators, for example, paddles or air jets, to separate objects [8].

1.1.4 Pick and Place Arms are Less Cost Efficient

There are a number of pick and place arm designs used in industry for sorting and assembly, including Selective Compliance Articulated Robot Arm (SCARA), articulated and delta arms [9]. Pick and place arms provide great flexibility as they can quickly move individual objects with very few constraints. This makes them ideal for removing occasional defect items from a group.

Pick and place arms can be used to build adaptive sorting systems that allow a wide variety of objects to be sorted.

S. G.Paulraj et al. [10] use a pick and place arm mounted on a mobile robot to sort recycling. The authors' approach uses visual classification to determine the type of recyclable material and then the robot collects it and transports it to a specific bin. This work shows that pick and place arms can be used for adaptive sorting. This work also shows the negative aspects of a pick and place approach that the robot is only able to manipulate one object at a time.

A downside to these systems is that moving individual items limits their throughput. To examine the throughput rate of pick and place sorters, I compared a delta robot arm to a robotic sorting machine and a mechanical sorting machine, shown by Table 1.1. The delta arm I include is a Hans Motor Industrial Delta Robot, it is capable of

Machine	HMI Delta	FS Tomato Sorter	LM Apple Sorter
Price (\$)	20000 (lowest estimate)	4000 (highest estimate)	1500 (highest estimate)
Throughput at 25% (50%) removal rate (s^{-1})	13.333(6.66)	6.83	4.90
Cost for throughput at 25% (50%) removal rate (\$s)	1500 (3000)	1171	306

Table 1.1: A comparison of three sorting methods. Throughput is given for both a 25% and 50% removal of objects from main line rate.

picking up 200 objects a minute. The robotic sorter I include is an FS robotic tomato sorter which weighs objects, washes them and applies a wax coating before sorting the objects into boxes with actuated buckets on a belt. The mechanical sorter I include is a Longer Machinery Mechanical Apple Sorter which passes apples between two rollers and depending on the size of the apples they fall through at different points. The results are compared by cost for throughput which I define as:

$$\text{cost for throughput} = \frac{\text{cost}}{\text{throughput}} \quad (1.1)$$

All values were taken from Alibaba, the prices are based on estimates provided on the site.

This table shows that as the number of objects that need to be removed from the main line increases the cost efficiency of sorting machines compared to pick and place arms increases. The delta arm is limited by the number of objects moved while sorting machines are usually limited by mass throughput. These different limiting factors make sorting machines even more cost-effective for lighter objects. The costs shown here do not include any kind of conveyor or sensors to use with the delta arm, while the sorting machines are mainly fully contained systems. This extra equipment may add significant extra cost or reduce the throughput of the pick and place system.

1.1.5 Peristaltic Tables Allow for Robotic Sorting on a 2D Plane

Soft robotics presents a new avenue for exploring and exploiting the use of the mechanical properties of a robot body. Soft robots show promise for parallel sorting in a way that crosses between the current domains of mechanical and robotic sorting.

Peristaltic tables are soft robotic systems with a potential for sorting applications [11]. Peristaltic tables are inspired by peristalsis in biological systems. Peristalsis refers to the expansion and contraction of a series of segments or cells to create moving, wavelike contractions in a hollow chamber [12]. These contractions force objects within the chamber to move along with the waves. This behaviour can be mimicked in machines to create peristaltic conveyors [13]. The term peristaltic motion refers to the more general movement of objects via wavelike motions created through expansion and contraction of

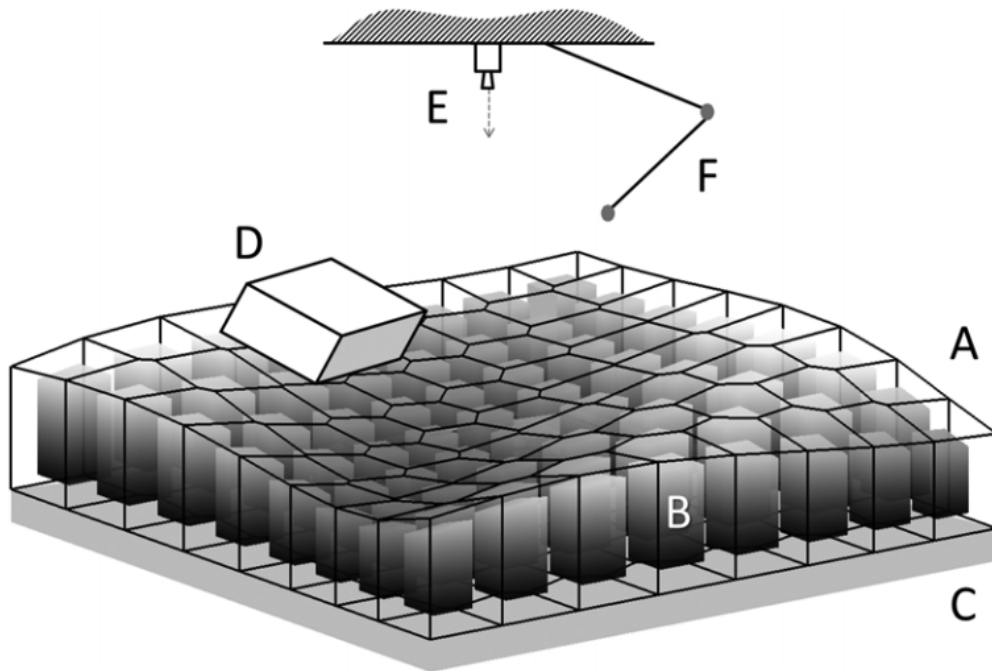


Figure 1.2: A diagram of a peristaltic table. (A) A flexible top layer. (B) Linear actuators. (C) Rigid base. (D) The object being sorted moves down the gradient created by the table. (E)-(F) Additional sensors and/or manipulators. Reproduced from M. Stommel and W. Xu [14] ©2015 IEEE.

cells, not necessarily within a chamber. When this motion is across a horizontal plane of cells the system is called a peristaltic table, shown by Figure 1.2. Peristaltic tables can manipulate objects on their surface by creating peristaltic waves with objects rolling or sliding under gravity along the gradients created by these waves. The expansion and contraction of the cells in a peristaltic table can also be useful for separating objects by lifting the area between them. These deformations can also allow for shaking clusters apart or vibrational sorting if a high enough frequency of expansion and contraction can be reached. Difficulties with this system may arise from the control of the potentially hundreds of actuators. The number of actuators and interactions between actuators can be too demanding to run in real time on one centralised controller. The distributed control algorithms found in modular and swarm robots would allow for control systems that scale inherently with the size of the table rather than relying on a single controller.

During my project, I aimed to develop an adaptive sorting system with the flexibility of robotic sorters while exploiting mechanical interactions and sorting multiple objects at once. I aimed to use a peristaltic table design for this sorter while also trying to address the existing issues with peristaltic table systems.

1.2 Problem Statement and Breakdown

Industries such as recycling centres and port authorities require the sorting of a stream of diverse objects. Current industrial sorting systems are too narrowly focused to handle these streams while new robotic sorters can not sort multiple objects at once, leading to

a limit on the throughput of the industry. Peristaltic tables can provide parallel sorting with the flexibility of robotic sorting. Four key issues with peristaltic tables, however, need to be addressed.

1. Current peristaltic systems are unscalable due to reliance on centralised systems. As the table becomes larger the number of actuators scales with the number of possible actions that can be taken by the table.
2. There are no available general simulators for peristaltic tables. This frustrates the implementation of any high-level, peristaltic control that relies on machine learning and slows the wider development of any peristaltic control. Without an available simulator researchers either need to design a new model for their system or go without a model.
3. Peristaltic control has only been generalised at a high level making the development of new peristaltic systems time-consuming. The algorithms for controlling peristaltic behaviour at a low level are usually bespoke and do not generalise.
4. Peristaltic tables have not been demonstrated moving multiple objects or sorting by different criteria. To have a more adaptive sorter, the hardware and algorithms of a peristaltic table need to be created such that they can handle a variety of objects and sort for multiple criteria.

1.3 Preview of Contributions

A brief summary of the contributions I made to the field of sorting robotics during this project:

- I have designed and demonstrated a novel modular robot, the Linbot. This robot can function as a cell in a peristaltic system allowing the system to use distributed algorithms. Distributing the computation of the system gives any peristaltic table made from the Linbots the potential to be scalable and parallel. The Linbot is not limited to this role and could easily be used in other assemblies or as a cheap tactile sensor node. This modular robot is designed for manufacturability and so is cheap and easy to construct [15].
- I have written a peristaltic table simulator that I have published as the python package PeriSim. This simulator is a general peristaltic table simulator and can be tuned to different systems. Previously there was a lack of available peristaltic table simulations, adding a barrier to developing peristaltic control in simulation. My simulator is designed to be easy to use and is available for other researchers. It overcomes the difficulties with modelling peristaltic behaviour using a radical envelope of noise that allows for controllers designed in a simulation to be robust to the changes in dynamics when moved to the real-world [16].
- I created a state-machine based algorithm for low-level peristaltic wave control [15]. This algorithm allows for controlling the speed, direction, and length of a peristaltic wave in a distributed manner. This algorithm also avoids collisions between objects moving on a peristaltic table [17].

- I have constructed and demonstrated a proof of concept, modular robotic, adaptive sorting system. My system represents a step towards sorting systems positioned between flexible pick and place systems and high throughput industrial sorters. I demonstrated my system sorting and conveying real agricultural products. My system provides a clear design for a future industrial sorting system to be built upon [17].

1.4 Publications

1.4.1 Directly Related to this Project

Linbots: Soft Modular Robots Utilizing Voice Coils, Ross M. McKenzie, Mohammed E. Sayed, Markus P. Nemitz, Brian W. Flynn, and Adam A. Stokes, *Soft Robotics*, 2019, 6:2, 195-205. See Section 3.2.

PeriSim: A Simulator for Optimizing Peristaltic Table Control, Ross M. McKenzie, Jamie O. Roberts, Mohammed E. Sayed, and Adam A. Stokes, *Advanced Intelligent Systems*, 2019, 1:8, 1900070. See Section 4.2.

A Modular Robotic Sorting Table, Ross M. McKenzie, Jamie O. Roberts, Mohammed E. Sayed, and Adam A. Stokes, *Soft Robotics*, In Review. See Section 5.2.

1.4.2 Other Contributions

During my project, I took an active interest in collaborating with others working on soft or multi-robot systems. These collaborations allowed me to compare and contrast approaches to system design and control, leading to improved approaches to my project. This work is not claimed as part of this project.

Integrating soft robotics with The Robot Operating System: a hybrid Pick and Place arm, Ross M. McKenzie, Thomas W. Barraclough, and Adam A. Stokes, *Frontiers of Robotics and Autonomous Systems*, 2017, 4, 39. During my previous masters project, I worked on a pick and place system with soft robotic elements. This work led me towards the idea that a robotic sorting system would benefit from the use of soft robotic components and encouraged me to seek solutions that overcame the limitations of pick and place systems. During my PhD project, I put the finishing touches on the results from this previous project and submitted the resulting paper for publication.

HoverBots: Precise Locomotion Using robots That are Designed for Manufacturability, Markus P. Nemitz, Mohammed E. Sayed, John Mamish, Gonzalo Ferrer, Lijun Teng, Ross M. McKenzie, Alfred O. Hero, Edwin Olson and Adam A. Stokes, *Frontiers of Robotics and Autonomous Systems*, 2017, 4, 55. In this work, I assisted with the development of an air table. This air table uses a flow of air to levitate the Hoverbots similarly to an air hockey table with a puck. The levitation bots can then use magnetic locomotion. I also contributed to the manuscript.

The Limpet: A ROS-Enabled Multi-Sensing Platform for the ORCA Hub, Mohammed E. Sayed, Markus P. Nemitz, Simona Aracri, Alistair C. McConnell, Ross M. McKenzie, and Adam A. Stokes, *Sensors*, 2018, 18:10. To better understand the use of distributed sensing for use with the Linbots, I collaborated on the development of a distributed sensor platform, the Limpet. I assisted with manuscript editing and the development of the communication systems.

Limpet: A Modular, Untethered Soft Robot, Mohammed E. Sayed, Jamie O. Roberts, Ross M. McKenzie, Simona Aracri, Anthony Buchoux, and Adam A. Stokes, *Soft Robotics*, Submitted. Stemming from the original collaboration on the Limpet I also collaborated on a system that combined the Limpet and with the actuators from the Linbots. Four Linbot actuators were used to create mobile modules carrying a Limpet. One design vibrated each Linbot actuator at different frequencies such that the robot moved via slipstick motion. Another design uses the actuators in combination with suction cups to adhere to and climb up a surface. In this work, I helped design and carry out the experiments as well as lending expertise on the module design.

1.5 Organisation of the Thesis

Chapter 2 provides a broad literature review around my project detailing much of the existing work that I considered during my project. Chapters 3, 4, and 5 consist of journal papers set within chapters that add information to the place of the paper within my overall project.

Chapter 3 details my development and demonstration of a novel modular robot called the Linbot. The Linbot is designed to function as an intelligent cell within an adaptive sorting system.

Chapter 4 details my work building a simulator for my sorting system to assist with developing control strategies. My simulator, PeriSim, is suitable for a variety of peristaltic table designs and I made it available for use in the research of others.

Chapter 5 details my use of the Linbots and PeriSim to build a modular robotic sorting system and my demonstration of the abilities and potential of the system.

In Chapter 6 I discuss future directions for my work and sum up my project. The appendices contain supplemental information detailing Linbot design and construction, supplemental information from the papers directly related to this project and copies of the papers resulting from other contributions I made outside the scope of this thesis.

1.6 QR Codes

This thesis contains QR codes to link to related online material. For example, the QR code below usefully links to this thesis.



Chapter 2

Literature Review

2.0.1 Building Peristaltic Tables

There are a variety of technologies that can be used to create peristaltic tables. In R. Hashem et al. [18] the authors build a peristaltic table consisting of a number of pneumatic cells that can be pressurised to expand. These expanding cells create the deformations in the table required for peristaltic motion. This system functions well but required a major pneumatic infrastructure to function, and uses central, global control which limits the scalability of the system.

A dissimilar approach to building a peristaltic table is taken by T. Tone and K. Suzuki [19]. The authors use a ferrofluid sandwiched between two thin sheets to create a surface that is deformed by magnetic fields. This surface is shown to be capable of moving and mixing droplets. This system also has limited scalability due to centralised control and the force output is highly limited making it only suitable for miniature scale systems.

Actuating adjacent peristaltic cells can cause the cells to deform in the plane of the table. This happens when the cells are linked by the surface of the table or when expanding cells push on one another. This effect was exploited by Z. Deng et al. [20] to create a table based on caterpillar motion. This motion isn't peristaltic as it does not rely on wave-like patterns, instead using sections of the table that move in the plane of the table rather than just normal to it. This caterpillar table is, however, highly similar to peristaltic tables in concept and construction. The caterpillar motion is shown by Figure 2.1. This system can move items which do not slide or roll well enough for peristaltic tables. The system requires flat objects to work but some hybrid control system could use both standard peristalsis and this system to move a wider variety of objects.

A highly lightweight and miniaturisable approach to expanding cells is shown by S. Kim et al. [21], who developed Kirigami actuators that can transform between a flat and cubic structure. These actuators are powered by very small SMA wires that provide tension across certain edges of the structure to cause the transformation from one shape to the other. This approach holds the potential for being used in a low-cost, low-weight peristaltic table but would likely not be able to scale up to enough force for industrial sorting.

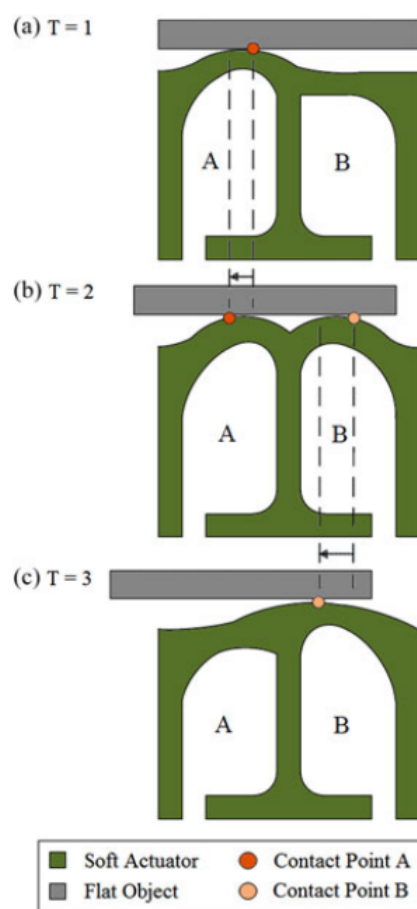


Figure 2.1: Process of object movement with inflating air chambers at different times. Based on caterpillar locomotion. Reproduced from Z. Deng et al. [20] ©2016 IEEE.

2.1 Sorting Algorithms for Swarm Robots

There is a lack of literature on modular robots being used for sorting. Swarm robots, however, are already being used for sorting in industrial settings. Swarm robots also use distributed algorithms for sorting and often rely on indirect control of objects for sorting. These aspects are similar to the algorithms needed to control a distributed peristaltic table.

2.1.1 Spatial Clusters of Swarm Robots

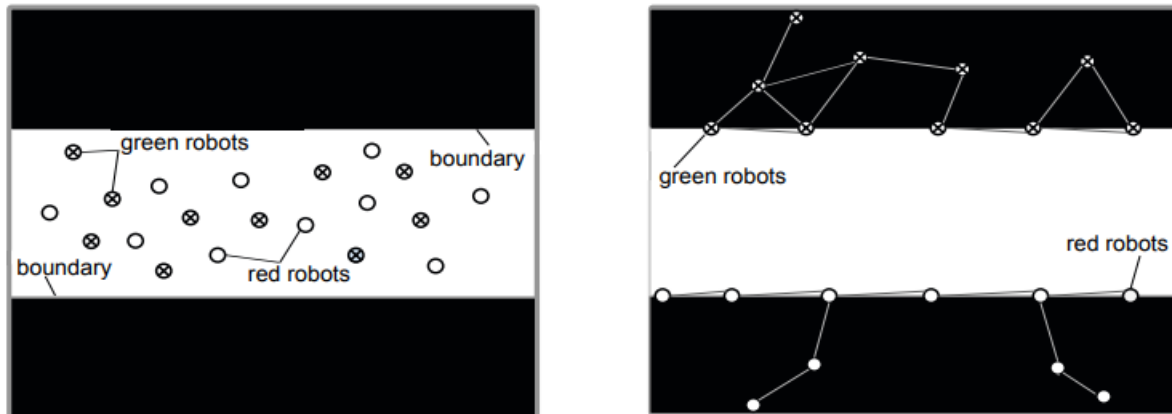
Robots with only local communications can perform system-level tasks through cooperation. This behaviour is demonstrated by N. B. Cruz et al. [22] with swarm robots that can perform spatial clustering. The robots used in their work only have local communication and knowledge of the world. Using local communication, the robots can cluster based on a defining feature. In this case, they sort into clusters with other robots assigned the same number. Cruz et al. demonstrate that the global effects can be created using local behaviours. These clusters form in an arbitrary position. This means that if the clusters needed to be in a certain position, as is often the case for sorting, the controller would need to also be able to handle moving clusters after they had formed.

2.1.2 Swarm Robots can Sort into Defined Areas

H. Ding and H. Hamann [23] demonstrate sorting with desired end zones. In their work, their robots sort themselves into separate end zones by class, as shown in Figure 2.2. Each robot knows its class and knows when it is in an end zone. The end zone that each class moves into is not prescribed and so the robots work out which zone their class is clustering in at runtime. The robots avoid clustering into the same end zone as other classes by forming barriers around the edge of a zone and stopping any other classes from crossing. Before an end zone has been fully surrounded by a barrier, it is possible for two barriers to start to form. If these barriers meet and are of different classes then the robots in the minority will leave the zone. The robots share knowledge about their neighbours to estimate how many robots are in their cluster. This work demonstrates the utility of using spatiality for sorting by forming barriers to keep the wrong robots out. Similar consideration of spatiality can be used in modular robots, by determining which robots are already best placed to serve, for example, as filters for passing objects. This work relies on the robots randomly wandering to find end zones, and therefore will lose efficiency as the size of the work area increases in proportion to the size of the end zones, and the speed of the robots or the object they're moving. Advantages may be found in combining arbitrary location clustering with desired-location sorting to allow for robots to start to cluster as they are searching for end zones.

2.1.3 Swarm Robotic Collaboration to Move Objects

Robots do not need to be directly mapped to the objects they are trying to sort and robotic sorting algorithms can use separate robots and objects. In the case of a peristaltic



(a) initial setup (red and green robots positioned within the white area)

(b) expected result (green robots share one black area and red robots share the other black area)

Figure 2.2: Spatial sorting swarm robots a) Initial setup b) Expected result. Reproduced from H. Ding and H. Hamann et al. [23] ©2014 Springer International Publishing Switzerland.

table the robots are immobile and require cooperation to move an object that can not be directly carried by any one robot. Algorithms overcoming this lack of individual ability is shown by the work by J. Chen et al. [24]. In this work, robots need to move an object larger and heavier than them and so must collaborate. The collaboration in this work is indirect and the robots were not sorting multiple objects. It does, however, demonstrate how robots can collaboratively move objects without picking up objects and carrying them.

Sorting with swarm robots requires that the robots are capable of locomotion. The need for individual locomotion increases the complexity of each robot. This complexity limits the number of swarm robots that can be applied to a task for a given cost.

2.2 Architectures for Modular Robots

Modular robots are similar to swarm-robots in their reliance on collaboration to perform tasks. Modular robots rely on a physical link between nearby robots and usually cannot move individually. Modular robots have been used to complete real-world tasks, including assembling into furniture [25]. However, many modular robots are not used in a task-focused manner and are instead aimed to replace the limbs and locomotion strategies in existing robotic applications. Existing modular robots can, however, provide potential designs for modular, robotic peristaltic table cells.

2.2.1 Modular Robots Share Characteristics with Swarm Robots

Modular robots share the distributed nature and limited abilities of swarm robots. Additionally, they do not need to be able to individually locomote, but are rather

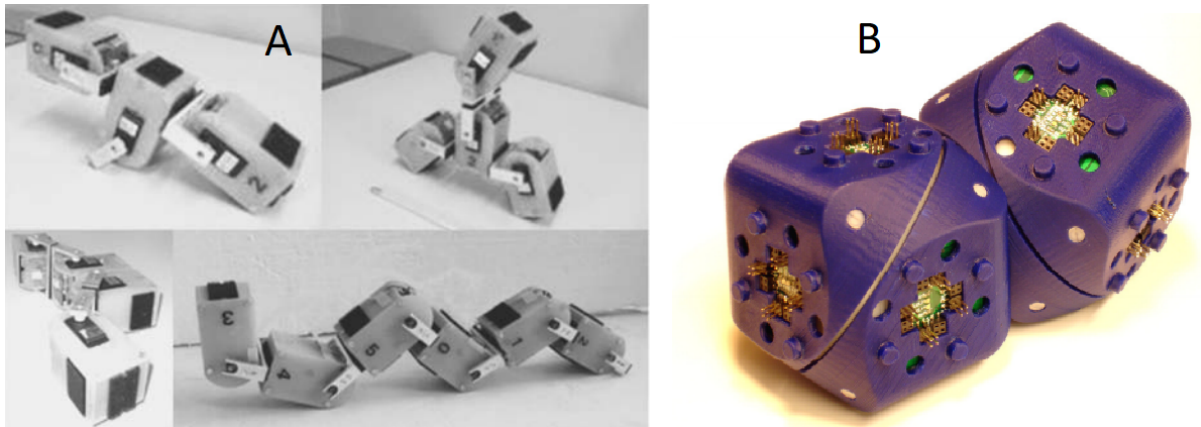


Figure 2.3: (A) Collectives of chain-like modular robots using YaMoRs. Reproduced from R. Moeckel et al. [27] ©2006 Emerald Publishing Limited. (B) A pair of modular robots with rotating halves using Molecubes. Reproduced from V. Zykov et al. [28] ©2007 IEEE.

physically connected to one another in a contiguous system. This connection means that the actions of robots in the collective affect the others. Using this interconnectivity modular robots can perform movements that require more degrees of freedom than each robot has individually, and so the individual robot can be simpler.

2.2.2 Rotational Joints Allows for Modular Structures

For example, some modular robots are built as (potentially branched) chains [26] [27]. These robots are linked together with 1 degree of freedom, actuated joints. The resulting chain can then perform movements which use all of the collective joints and have as many degrees of freedom as there are robots. Examples of chain-like modular robot collectives are shown by Figure 2.3. Other modular robots are built from two rotating halves [25] [28], shown by Figure 2.3. This rotation allows them to change the angle between their neighbouring robots. These modular robotic designs would require mechanical adaptation to produce linear motion for peristaltic or vibrational sorting.

2.2.3 Robots can Move Around their Collectives

M-Blocks [29] are modular cube robots that can rotate their entire body using an internal flywheel. Their rotation allows them to move around the other robots in the collective and allow the collective to change its morphology and locomote. This approach could be used for sorting as individual blocks could push objects individually or through coordination. In this setup, however, the robots would behave like swarm robots as their modular nature is not being used and other swarm options have more efficient locomotion.

2.2.4 Expanding Modules can Perform Tasks in a Collective

A. Vergara et al. [30] demonstrate “soft, modular, robotic cubes” that can form and reorganise their collectives. These cubes attach to one another with embedded permanent

magnets and have pneumatic bladders on each size which allows them to expand. The robots can detach from one another by expanding their connected side which separates their magnets. The robots ability to grow allows the combined robot to create peristaltic waves and rearrange itself. This approach would allow for the use of the collective to sort objects via peristaltic waves and vibration. However, these cubes require external sources of pneumatic force, sensing, and computation so cannot be used for distributed sorting algorithms.

D. Zappetti et al. [31] have created tensegrity based soft modules for robots. Tensegrity describes having multiple elements connected under tension and compression. These tensegrity modules have been actuated by an internal motor that shortens a tendon connecting the top and bottom halves of the modules. When compressed by the shortening of these tendons the modules expand radially. They use these modules to create a soft, worm-shaped robot that moves via a peristaltic wave passed along its length. Theoretically, these modules could have onboard power and computation. However, these authors note that the system could be improved with an integrated actuation mechanism as the motor used to shorten the tendon resulted in slow actuation and limited the change in shape of the motor.

2.2.5 V-SPA Modules use a Vacuum Source to Bend and Actuate Linearly

Work by M. A. Robertson and J. Paik [32] shows that multiple actuators can be combined into a single modular robot, shown by Figure 2.4. Their robots are Vacuum-powered Soft Pneumatic Actuator (V-SPA) modules as they consist of three V-SPAs arranged in a triangle connecting two rigid halves. The V-SPAs compress when a vacuum is applied to their control line and the separate control of three of these actuators allow the V-SPA module to bend and expand along its central axis. V-SPA Modules would work for a peristaltic system as they actuate linearly and their bending motion could allow for extra motions not possible with a linear only sorting table design. V-SPA modules have onboard computation limited to controlling the three actuators based on a control signal. This limit means that higher level control would need to come from an external source or require redesigning the electronic system of the V-SPA modules. These use of pneumatic power in the V-SPA modules means that they would also require an external vacuum source and means that each actuation takes 1.5s.

2.2.6 Voice Coils can Be Used for Linear Actuating Modular Robots

Soft modular robots built around an integrated actuator are demonstrated by Markus P. Nemitz et al. [33] using a voice coil actuator with a hybrid body that functions as a spring. In this work, the modules are also assembled into a locomotive worm-shaped robot, shown in Figure 2.5. They also demonstrate a fully untethered module with onboard power and computation. These soft modules have the advantages of high-frequency actuation as well as their choice coil functioning as an antenna and loudspeaker. They create the body of the actuators using 3D soft lithography and 3D printing which are time-consuming methods and limit the number of robots that can be built for a collective. They could be improved to be more manufacturable and have

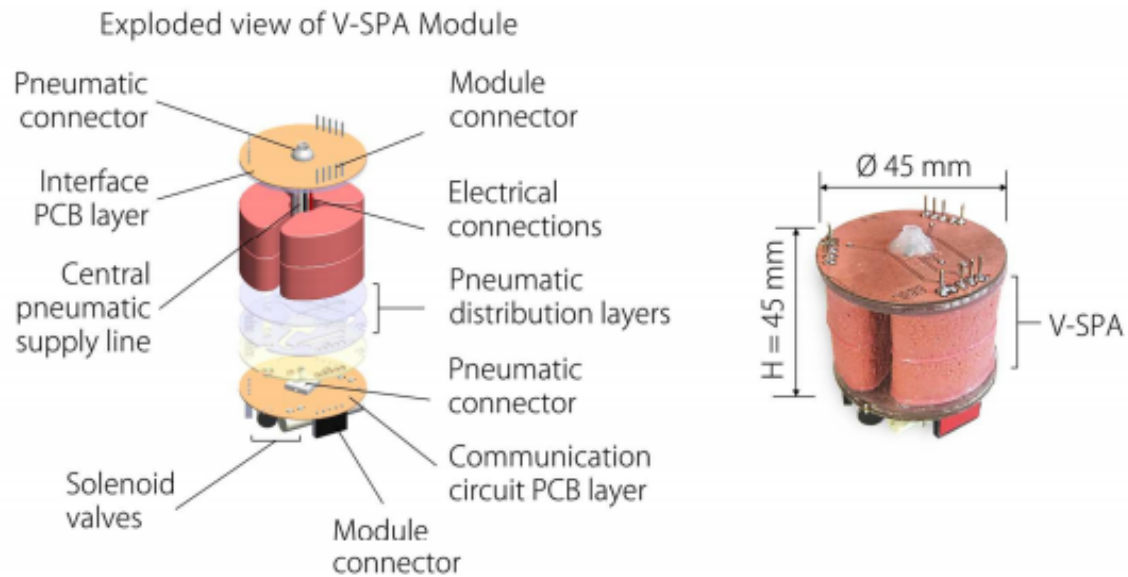


Figure 2.4: An overview of a V-SPA module. Reproduced from M. A. Robertson and J. Paik [32] ©2017 The Authors, some rights reserved; exclusive licensee American Association for the Advancement of Science.

greater computing power.

2.3 Learning System Dynamics for Collective Control

The effect of the interactions between many robots in a collective is difficult to predict, this makes standard robotic controllers difficult to write and can lead to unexpected behaviour [34]. Methods of control that are based on these interactions can be used to produce more reliable results and desirable behaviour. Machine learning can be used to control systems whose dynamics are difficult to understand analytically. This machine learning model then needs a suitable controller architecture to control the robots.

For a machine learning approach I need to take into account two attributes of my system:

- The optimum behaviour for a collective to complete a task is unknown. Therefore, unsupervised methods must be used.
- The controller needs to be able to learn temporal relations. This allows behaviours that contribute to non-immediate reward to be learned.

2.3.1 Evolutionary Algorithms can Optimise a Controller Without Prior Knowledge of the System

An Evolutionary Algorithm (EA) is a biologically inspired optimisation technique. In general, an EA uses a population of semi-random solutions and selects the fittest (best performing) members as a basis for the next generation of the population. The measure

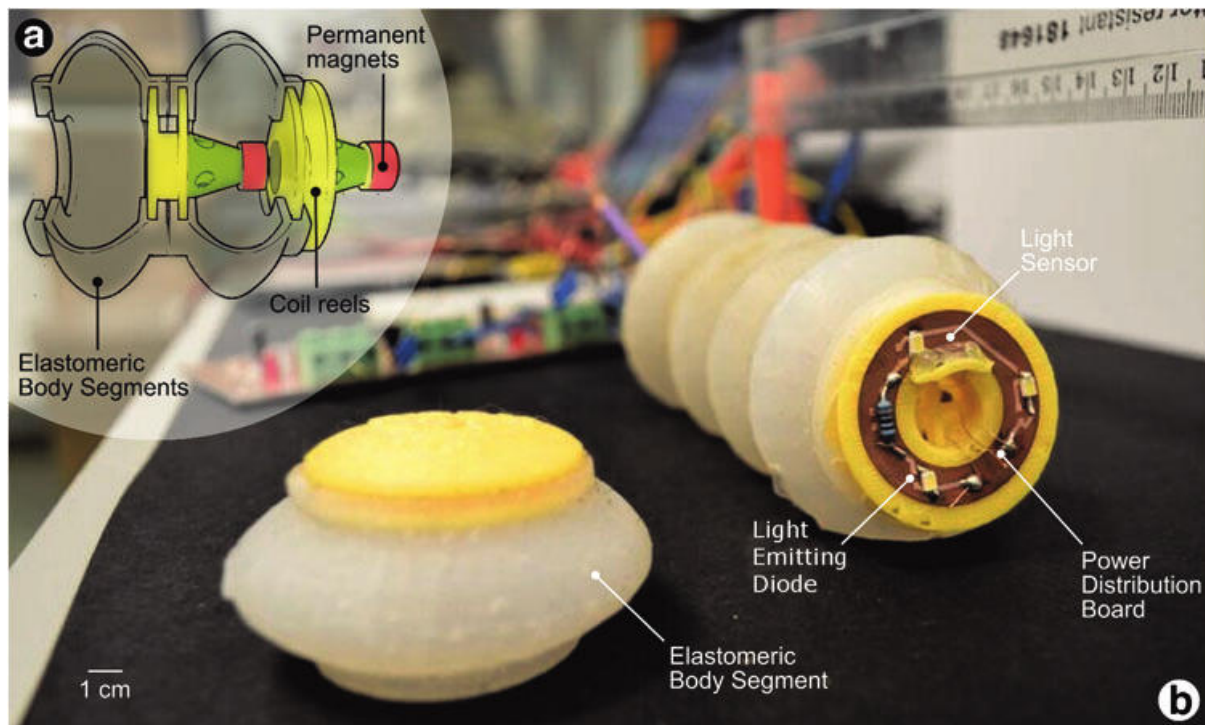


Figure 2.5: Overview of the Wormbot. (a) the voice coil actuators (b) the assembled Wormbot. Reproduced from M. P. Nemitz et al. [33] ©2016 The Authors; Mary Ann Liebert, Inc.

of fitness and the system for creating a new population differs between types of EA and will usually involve both random perturbations and mixing of the fittest solutions. EAs can learn to leverage the interactions between robots.

Wenguo Liu and Alan FT Winfield [35] use probabilistic finite state machine architecture and a Genetic Algorithm (GA) to optimise foraging swarm robots in a simulation. The genetic code of their robots determines the parameters used in adapting the state machine to certain cues in the environment and from other robots. Their work showcases how GAs can be used to make optimising a large search space tractable and how this can be used for learning in multi-robot systems. The authors' work only deals with simulated robots which means that it may not work with real-world robots. Their approach does allow their robots to adapt in real time, another approach to real-time adaptation could come through an online GA.

2.3.2 Reinforcement Learning Can Be Used with Modular Robots to Learn Gait

Reinforcement Learning (RL) uses repeated trials to learn expected rewards for certain actions in certain situations. Over time these rewards are propagated backwards through previous actions taken to lead to the reward. In work by D. J. Christensen et al. [36] RL is used to make a modular robot with unknown morphology walk. Their approach shows that reinforcement learning can work with modular robots.

2.3.3 Distributed, Scalable Learning

Neither GA or RL approaches are distributed in their basic forms. Both approaches use system-wide information to optimise control in a manner that would be impossible in a system where robots each only have local information, such as a modular robotic collective. Simon Jones, et al. [37] adapt GAs for a distributed application by having each robot run a GA with some passing of genetic information between robots. This approach allows for optimisation of individual robot behaviour while sharing some useful collective-wide adaptations. RL can also be performed across multiple robots. Hongliang Guo and Yan Meng [38] use a similar approach of having individual optimisations with some shared information between robots. To perform either approach using only modular robots requires that the robots have substantial on-board computational resources. For low-cost modular robots a new approach will need to be taken, or much of the optimisation needs to be carried out in a simulated environment before moving to real-world robots. For optimisation in simulation both GA and RL approaches have been performed in a scalable manner that allows for parallelisation across multiple computers [39] [40].

2.3.4 State Machine Genetic Algorithms are Suitable For Discrete Environments

GAs involve using a genetic code that contains the action a robot should take in each situation [41]. This genetic code is usually a series of numbers, the solution searches for a particular position in the code and uses the number in deciding the next action to be taken. These are useful when there are a finite number of discrete environmental states that a robot can be in so each can be assigned a gene. States can also be set to correspond to ranges of sensor values. This means that even a continuous environment can be broken down into discrete states. This discretisation, however, losses some sensor data and increases the number of parameters that need to be tuned by hand or through machine learning.

2.3.5 Evolved Behaviour Trees can Create Human Readable Controllers

Behaviour trees are similar to decision trees with extra nodes that encode behaviour other than decisions. Decision trees are common in video game AI and Chong-U Lim et al. [42] details the use of EAs to evolve a behaviour tree for the game DEFCON that outperforms the AI which was coded by the developer. This work demonstrates how behaviour trees can be used with EAs. Behaviour trees can present controller behaviour in a more human readable way than state action pairs.

The use of these Evolved Behaviour Trees (EBTs) for robotics can be seen in work by Simon Jones et al. [43] where a swarm of foraging Kilobots use EBTs to control foraging behaviour. This controller is evolved in simulation and yields a human-readable control scheme. It also functions in real-world experiments, although with reduced performance.

2.3.6 Distributed Hormone Control Enables Robust Communication

H. Hamann [44] use biologically inspired hormone control to leverage the local communications of collective robots for control. The work was carried out in simulation and uses locally communicating agents to simulate hormone reaction-diffusion. The hormone level of a robots then affects its actions. In this work they use this system to control the gait of a robot made of multiple homogeneous modules. This system works well with robots that communicate locally and doesn't require the robots to have unique identifiers. The diffusion and decay rates of the hormones and their effect on the robots were evolved via GAs. This work required a large number of iterations to learn desirable characteristics, this suggests that the characteristics are highly tuned to the simulated environment that they were optimised in. This level of tuning would suggest a large "reality gap" when applying the controller to a real-world robot.

2.3.7 Peristaltic Control Using a Probabilistic Automaton

A method to control the waves of a peristaltic table has been detailed by M. Stommel and W. Xu [14]. The authors' implementation uses a set of predefined actuation behaviours and divides all of the possible configurations of the system into discrete states. Through multiple experiments, the controller learns the probability density of the next state based on the current state and the chosen action. This allows an optimal path through the individual states to be calculated which takes into account the noise in the system. This system is independent of the true hardware setup and should be robust to objects behaving differently to the same peristaltic actuation.

Control through a probabilistic automaton has been shown to work on a real world system [45] based on inverted caterpillar locomotion [20], the system is discussed in Section 2.0.1. In the experiment a flat object is translated and rotated to reach a goal while avoiding an area on the table. This shows the feasibility of using this approach for sorting objects via a peristaltic table. This does represent a less stochastic problem than a conventional peristaltic table as the inverted caterpillar motion doesn't use slipping or rolling to move objects around. This approach also requires completely centralised control and so it would need to be adapted for distributed control across multiple robots.

2.4 Overcoming The Reality Gap

This reduced performance when moving controllers from simulation to the real world is called "the reality gap". This effect is caused by the inherent differences between a simulation and the real world. These differences usually occur in the effects of robot actions or the sensory data a robot receives. If the controller relies on the accuracy of one of these aspects then its performance will suffer when it is used in the real world.

2.4.1 Human Editing to Adapt a Controller to the Real World

Further improvement could come from human editing of evolved solutions. A masters thesis by K. Y. W. Scheper uses EBTs to produce editable controllers for the Delfly Explorer, a flapping-winged robot [46]. This work suggests that human edited evolved

solutions can produce better results than both the evolved solutions and human-designed solutions. This technique requires EBTs as other EA systems such as artificial neural networks and finite state machines produce solutions that are not inherently readable and so can be difficult to improve through editing in their basic forms. A similar human ability to edit is available when the physical form of a robot is designed through EAs, although the dynamics may be harder for a human to predict.

2.4.2 Real World Validation Steps can be Used to Improve Simulations

P. Abbeal et al. [47] use a mixture of simple simulation and real-world validation steps to bridge the reality gap. Their technique uses model-based reinforcement learning, meaning that their controller has access to a simple model of the world to predict the results of actions. It uses these predictions and their rewards to determine its actions. One of the authors' examples is the simulation of an RC car whose model doesn't include drift or slip and is fully deterministic. The model used by the controller is then also used as a simulation in which to optimise the controller. Using the evolved controller they then do an experiment on their real-world system and record how the actions taken by the controller affect the next state. They use this data to find a bias term to change the model to make it more in line with the real world results. They then repeat the optimisation, experiment process with their biased model, and therefore simulation until it displays optimal real-world behaviour. Even though there were still errors in the trajectory of their RC car at the end of the process the technique allowed for a controller that performed far better than one purely evolved in simulation. Their work does rely on each robot being able to model its own dynamics, for collective robotics these models will need to be simpler due to the limited power of each robot and the authors state that more complex models produce better results. Work by A. Nagabandi et al. [48] suggests that collective robots could learn the dynamics of their system using an Artificial Neural Network (ANN), which can be easier to run than other models. To achieve this their robot required a constant data link to another computer to train the ANN .

2.4.3 Adding Noise to Simulation Parameters Creates Robust Controllers

N. Jakobi [49] improves on the idea of building noise into a simulation by suggesting that the underlying behaviour of a simulation should be different at each run. This is achieved through perturbing the parameters used in each simulation run, these parameters include the strength of gravity, the time-step, the abilities of each robot, etc. This leads to controllers that must be robust to small changes in the underlying dynamics of the world, and so can cross the reality gap, but also are not designed to make use of more noise than is realistic at runtime. The authors note that this system can work with minimal simulations where only behaviourally relevant interactions between the robot and environment are taken into account. Careful design is required to ensure that the proper interactions are modelled. The correct amount of variance in the underlying parameters can also be difficult to determine and so may require a trial and error approach with multiple evolutions. This approach would work well for my project as no peristaltic table simulations currently exist and it allows for a relatively simple simulation to create robust

controllers. Building a very simple simulation will both save on development time and allows for a lower computational overhead for optimisation.

Chapter 3

Design and Fabrication of the Linbot

3.1 Introduction

There are currently no modular robots designed for use in peristaltic tables or even for sorting more generally. By creating soft modular robots that functioned as peristaltic cells I aimed to build a sorting system that could take advantage of both robotic and mechanical interactions for sorting. My design was inspired by the voice coil system used in the Wormbot, see section 2.2.6. This design uses a magnet and coil of wire connected by a spring to create extension and contraction motions.

3.1.1 Mechanical Design

- I designed a spring to function as the main body of the robot and the connection between the magnets and the coil. I made the springs through Kirigami with flexible plastic. Kirigami is the use of cut and folded sheet material to create 3D objects. The design allows for a number of springs to be made in a strip on a laser cutter. Tearable sections allow the individual springs to be removed from the strip easily. I also incorporated barbed tabs along either side of the spring that slot into the PCB and the cap. When the spring is folded into its 3D structure its connection to the PCB and cap mean that it holds its shape. Its 3D shape is similar to a Chinese paper lantern with ten bent legs holding the two end cylinders at a distance. These bent legs can be further bent or stretched but will return to their equilibrium state, this is what makes this structure function as a spring. This design makes assembling the robot by hand much easier and allows for them to be disassembled without damage. The final kirigami design is shown by Appendix A.
- I designed the robot to have two rigid end caps to connect the magnet and coil to the spring. In order to keep all manufacturing to 2D processes I designed one end cap to be made of laser cut plastic. I decided that the PCB itself would also function as an end cap, as it is rigid and I could cut it to the right shape.
- I designed cylindrical holders for the coil and magnet that connect to the end caps. The distance between magnet and coil controls the maximum force that could be produced. I designed the coil and magnet holders such that the top of the magnets is at the same level as the bottom of the coil while at rest. This design maximises force output from the rest position.

3.1.2 Electronic Design

- Modular robots require on board decision making. I added a microcontroller to the Linbots so that I could program them with distributed control algorithms.
- Markus Nemitz et. al. [33] showed that voice coil actuators can be used as an antenna for magnetic communication. By using the coil for both for actuation and communication I could reduce the overall cost of the Linbots. I designed the Linbot with an LC oscillator [50] to provide on-board power and control for the communication. The LC oscillator uses three coils. Two coils have 12 turns, the resonant coil and the trigger coil, and the main, actuation coil has 200 turns. An LC oscillator produces an oscillating current through the resonant coil when it is turned on. Data is transmitted by the oscillator being on for 1 and off for 0. The magnetic field produced by these oscillations is magnetically coupled from the resonant coil into the main coil. The resulting magnetic field produces an oscillating current in the main coil of other nearby robots. The receiving circuit to decode these transmissions is made from two type-A amplifiers, an envelope detector and a comparator. This circuit recreates the transmitted data.
- In order to allow for both external sensing and proprioception I designed the Linbots to function as tactile sensors. I designed this sensing based on work by T. Paulino et al. [51], in which the authors used permanent magnets softly attached to a hall effect sensor to measure force. A 3-axis magnetometer was the only extra component that I needed to add to allow the displacement between the two halves of the robot to be sensed. I programmed the Linbots to track the displacement between the magnetometer (attached to the PCB) and the permanent magnets (attached to the other side of the spring).

3.1.3 Completed Linbots

I characterised the behaviour of the Linbots and demonstrated their ability to produce useful work through a conveyance and a sorting task. I submitted this work to the journal *Soft Robotics* where it has been published. Scan the QR code below for a short video summary of the paper.



3.2 Linbot Publication



Linbots: Soft Modular Robots Utilizing Voice Coils

Ross M. McKenzie,^{1,2,*} Mohammed E. Sayed,^{1,*} Markus P. Nemitz,^{1,3} Brian W. Flynn,¹ and Adam A. Stokes¹

Abstract

Robots performing automated tasks in uncontrolled environments need to adapt to environmental changes. Through building large collectives of robots, this robust and adaptive behavior can emerge from simple individual rules. These collectives can also be reconfigured, allowing for adaption to new tasks. Larger collectives are more robust and more capable, but the size of existing collectives is limited by the cost of individual units. In this article, we present a soft, modular robot that we have explicitly designed for manufacturability: Linbots. Linbots use multifunctional voice coils to actuate linearly, to produce audio output, and to sense touch. When used in collectives, the Linbots can communicate with neighboring Linbots allowing for isolated behavior as well as the propagation of information throughout a collective. We demonstrate that these collectives of Linbots can perform complex tasks in a scalable distributed manner, and we show transport of objects by collective peristalsis and sorting of objects by a two-dimensional array of Linbots.

Keywords: soft, modular, voice coils, Linbots, robots, soft robotics

Introduction

Large robotic collectives allow for robust behavior

THERE HAS BEEN major interest in modular robotic systems over the past years. This interest is due to the hypothesis that multiple low-cost units are more robust than a single, advanced robot.^{1,2} Modular robot collectives can be reconfigured for different tasks,^{1,3-5} setting them apart from monolithic robots. Collectives of modular robots have greater capability than the sum of their parts.⁶ This increased capability allows modular collectives to be used for tasks that are not possible with monolithic robots such as reconfigurable furniture⁷ and modeling cell collectives.⁸ Although modular actuators can also be combined into a single system for tasks such as controllable surfaces,⁹⁻¹¹ collectives of modular robots have the advantages of reconfigurability and inbuilt sensing and computation.

The capability of a modular robot collective is dependent on the number of units within it.¹ The size of a collective

robotic system, however, is often limited by cost, with larger groups sacrificing the functionality of individual robots to keep costs low. Creating low-cost robots with more capabilities is one of the key challenges in collective robotics. In this work, we developed modular, soft robots, demonstrated in Supplementary Video S1 (Supplementary Data are available online at www.liebertpub.com/soro), designed for manufacturability and which use multifunctional components to reduce hardware costs.

Multifunctional hardware can lower costs

The use of multifunctional hardware is demonstrated by the Kilobot.¹² The Kilobot is the lowest cost (~\$15) robot currently available. It uses an actuation mechanism based on cheap vibrational motors. It also uses a single infrared radiation (IR) sensor for both proximity sensing and communication. Through pulsing its IR transmitter, a Kilobot is able to communicate with others and by measuring the intensity of

¹Scottish Microelectronics Centre, School of Engineering, Institute for Integrated Micro and Nano Systems, The University of Edinburgh, Edinburgh, United Kingdom.

²EPSRC CDT in Robotics and Autonomous Systems, Edinburgh Centre for Robotics, Edinburgh, United Kingdom.

³Department of Computer Science and Engineering, University of Michigan, Ann Arbor, Michigan.

*These authors contributed equally to this work.

incoming IR communications. It is also capable of detecting how close other Kilobots are to it. This use of a single piece of hardware for multiple functionalities reduces the cost of robots.

Building a robot with an extensible architecture allows extra sensors to be easily added to the robot.¹³ This architecture allows the user to define the required abilities of each robot and keeps the cost-to-volume ratio balanced with respect to the task at hand. The base of the robot can then be repurposed for different tasks.

The HoverBot¹⁴ uses a Hall-effect sensor for both odometry and navigation.¹⁵ This swarm robot is made of a single printed circuit board (PCB) with electromagnetic coils and uses an air table with embedded permanent magnets for low-cost locomotion. Absent magnets on the air table create magnetic landmarks that can be detected by the HoverBots. This dual use of a Hall-effect sensor allows the number of required sensors and the cost of each HoverBot to be reduced.

Our Linbot uses a voice coil system for actuation, audio output, sensing, and communication. This voice coil system is based on the Wormbot.¹⁶

Voice coils for actuation, communication, and sensing

Voice coil systems are electromagnetic systems that are based on the same fundamental principles as loudspeakers; their method of actuation is detailed in Supplementary Figure S1. The voice coil system consists of an actuating electromagnetic coil and a permanent magnet, which are attached together by a soft body that functions as a spring. The electromagnet can repel or attract the permanent magnet and this force stretches or compresses the body of the robot, creating linear motion. The voice coils are hybrid hard/soft systems. Hybrid systems can take advantage of rigid components while still being able to partially conform to their environment.¹⁷

Voice coils have been shown to be a useful actuation system for soft robotics.¹⁶ The voice coil systems are multifunctional components that demonstrate different behaviors over a range of different control signal frequencies. In addition to actuation capabilities at low frequencies, the voice coils can function as loudspeakers at audible sound frequencies and can be used for communications at higher frequencies.¹⁶ This communication is based on inductive data transmission.¹⁸

We also utilize the permanent magnets in our voice coil system for sensing. The soft body of the robot deforms when it is actuated or when it is subjected to external force and we measure this deformation by monitoring a three-axis Hall-effect sensor, which responds to the relative position of the permanent magnets that are embedded on the top of the robot. This approach allows us to perform both proprioceptive sensing and tactile sensing.¹⁹

Building soft modular systems with voice coils

The simple linear action of voice coil systems is similar to the action of myocytes (muscle cells). Myocytes can only perform a simple action, contraction, and respond to simple inputs, signals from a nerve.²⁰ Many of these simple pairs of myocytes and nerves, placed at different positions and orientations, can produce complex actions such as prehensile movement or skeletal locomotion. Stacking units in this way can produce useful behavior for robotic applications.^{21–24} In this article, we provide two examples of stacking: first, peristaltic locomotion, which has been a major interest for soft robotic researchers.^{25–28}

Second, we rearrange the units used for peristaltic locomotion into a grid to create a peristaltic table that is capable of moving objects over the surface.^{29,30} Two-dimensional (2D) matrices of actuators have been created by Kim *et al.*,⁹ Stanley *et al.*,¹⁰ and Follmer *et al.*,¹¹ and these arrays could be adapted to allow a system designer to make a peristaltic table.

In this article, we have combined both computation and sensing into actuators that we arrange in a 2D array to create a fully distributed and modular peristaltic table.

To allow voice coils to function similarly to nerve/myocyte pairs, they need onboard computation to control their behavior. By including this functionality, we have created linear modular robots that can be stacked together. We have developed low-cost robots—Linbots—that can be configured to produce different forms of useful behavior. Our Linbots are capable of communication, actuation, sensing, and proprioception all through their central voice coil system.

Linbot Design

Design of the hardware

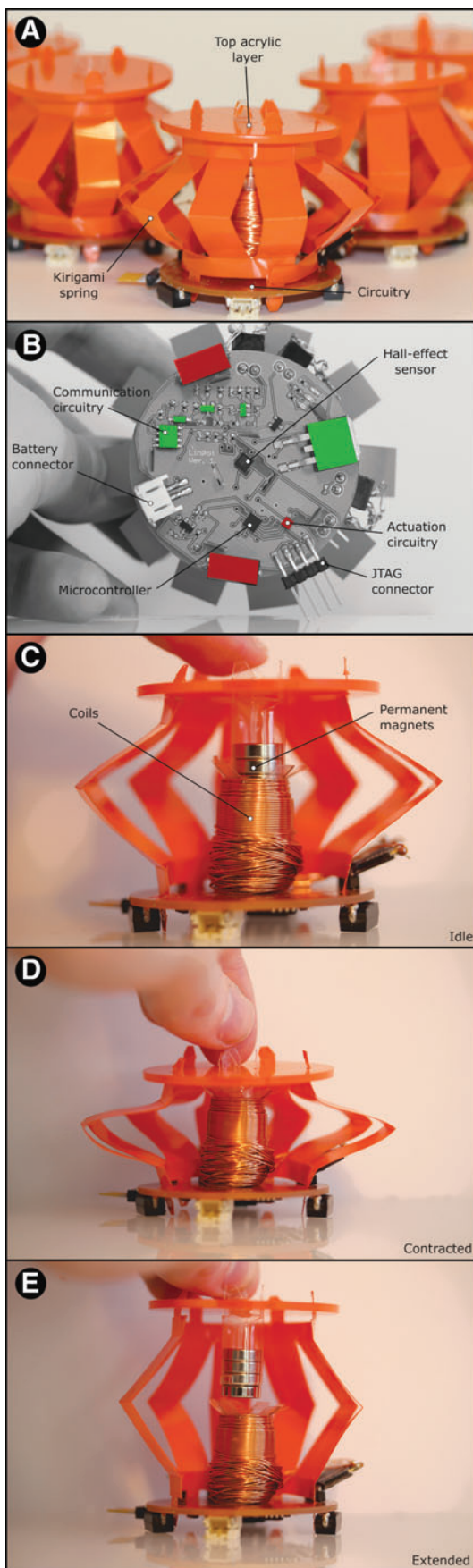
Figure 1A shows an individual Linbot unit. The main body of each Linbot is made up of the voice coil system, which includes the following: electromagnetic coils wound around a reel; permanent magnets embedded in a holder; and a spring consisting of connected bent legs resembling a Chinese lantern. A sketch of the Linbot, including its main components, is shown in Supplementary Figure S1. The magnet holder is attached to the spring via a circular, acrylic top layer. The coil reel is attached to the spring via the PCB, which serves as the bottom layer.

The Linbot can be extended or contracted axially from its rest position, depending on the polarity of current applied to the electromagnetic coil in the voice coil system. A sketch of the actuation mechanism is shown in Supplementary Figure S1. The sensing capability of the Linbots is achieved by a combination of a three-axis Hall-effect sensor incorporated in the PCB and the permanent magnets in the voice coil system. The Hall-effect sensor can track the motion of the magnets in three dimensions. This allows the Linbot to function as a tactile sensor since a change in displacement due to the addition or removal of objects on the Linbot can be easily detected.

Design of the voice coil system. We made the magnet holder, reel, and spring from acetate sheets using kirigami. Kirigami involves cutting a pattern out of the sheets and folding it into the desired configuration, the 2D patterns are shown in Supplementary Figure S2. We use kirigami as it allows our components to be low cost and highly manufacturable.³¹ The electromagnetic coils consist of two 12-turn coils used for the transmission circuit, and a larger 200-turn coil used for the actuation and receiver circuit. A circuit diagram of the coils is shown in Supplementary Figure S3. Supplementary Figures S4 and S5 show the block diagram and circuit schematic of the transmission and receiver circuits.

We used permanent neodymium magnets in our voice coil system. The internal components of the voice coil system are shown in Figure 1C. Supplementary Figure S6 shows a labeled picture of the custom-built coil-winding machine used for producing the actuation coils of the Linbots.

Design of the control electronics. We designed a fully integrated PCB incorporating a transmission circuit, receiver



circuit, microcontroller (STM8S), voltage regulator, H-bridge (DRV8837), and Hall-effect sensor (MLX90393) as shown in Figure 1B. The PCB schematic is provided in Supplementary Figure S7. We designed each Linbot to use a single PCB for control, sensing, and communication. The PCB is powered by a 450 mAh 7.4 V lithium polymer battery. Our communication system utilizes electromagnetic induction for data transmission, where the transmission coil generates an alternating magnetic field that induces a voltage in the receiver coil. Further information on the electronic design is detailed in The Electronic Design of the Linbot section of Supplementary Data.

Design of the control software

We wrote the control software in C and used standard peripheral libraries from ST,³² including I²C libraries for the Hall-effect sensor and Universal Asynchronous Receiver/Transmitter (UART) for communications. The Hall-effect measurements of magnetic field are used to calculate the displacement of the top half of the Linbot from its rest state.

We use the standard UART protocol³³ for magnetic communication to take advantage of the libraries provided by ST. The UART protocol uses a high idle line, which is pulled low at the start of a message. However, with the Linbot system, if two robots have their transmitters switched in, then all communications will be blocked. Therefore, we need our robots to switch their transmitters off when they are not sending any data. This requirement can lead to an extra byte, with a 0 value, being received at the end of each transmission. To avoid the extra bytes affecting the received data, we check for and remove these erroneous bytes.

Experimental Design

Design of the experiments to characterize an individual Linbot

This section discusses the design of the experiments used to demonstrate the capabilities of a single Linbot.

Quantifying output force. To evaluate the output force of the Linbots, we designed a controllable experimental setup with a scale, ruler, and clamp, shown in Figure 2A. Before starting the experiment, we zero the scale with a Linbot, a battery, and a ruler on it. We programmed the Linbot to extend periodically with maximum force. We used a clamp from a retort stand to restrict the height of the Linbot to different relative lengths. Since each of our hand-folded springs has a different rest length, we compare the force to the relative length rather than the absolute length. The relative length is given by the following:

$$x_r = \frac{x}{x_0}, \quad (1)$$

where x_r is the extension, x is the absolute length, and x_0 is the rest length. x_0 is measured before starting the experiment. We

FIG. 1. System overview. (A) A side view of a Linbot showing the kirigami spring and top acrylic layer. (B) A Linbot PCB. (C) A cutaway of a Linbot. The coil interacts with the permanent magnets and either pulls or pushes the kirigami spring. The circuitry is embedded on the *bottom side*. (D) A Linbot contracting. (E) A Linbot extending. PCB, printed circuit board.

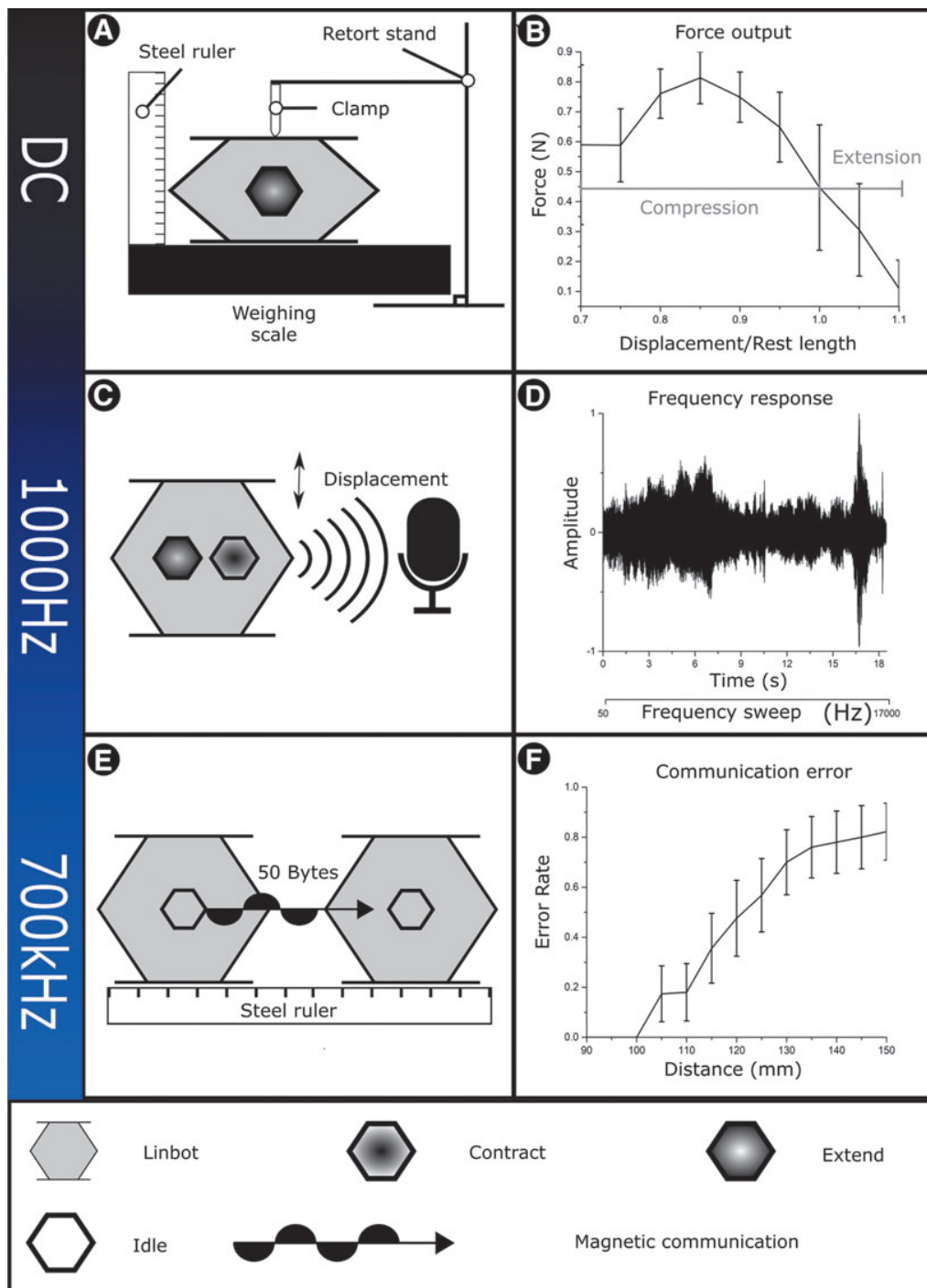


FIG. 2. Frequency-dependent functionality. **(A)** A schematic of our experiment to quantify the force output. This experiment involved control frequencies of hertz or lower. **(B)** The force output of the Linbot against relative length. The force output shown in the figure is a combination of the electromagnetic and spring forces of the voice coil. This shows that the Linbot has the highest force output around its rest length. **(C)** A schematic of our frequency response experiment. This experiment involved control frequencies of tens of hertz up to tens of kilohertz. **(D)** The sound wave produced by the Linbot during the last 18 s of a frequency test, shown in Supplementary Video S3. **(E)** A schematic of our experiment to evaluate the communication between Linbots. This experiment involved control frequencies of hundreds of kilohertz. **(F)** The byte error rate. This shows that the Linbots can communicate without error with up to 100 cm between their centers.

increase the extension in steps of 0.05 from 0.65 to 1.1; these extensions represent the full range of motion for the Linbots. Using relative length will introduce some error to the experiments as the electromagnetic force from the voice coil will depend on absolute length. The measured mass on the scale was used to calculate the force output of the Linbot at each extension. This output force is a combination of the spring force and electromagnetic force.

We repeated the experiment on a sample of five different Linbots as a screening experiment; this sample size allowed us to measure more than half of the available Linbot population (nine in total). We recorded the average force from these experiments and standard deviation at each extension. One repeat of this experiment is shown in Supplementary Video S2.

Frequency response analysis. The primary function of the voice coil system is linear actuation at low frequencies, but can also be used across a wide frequency range to provide audio output or communicate with neighbors. In this experiment, we programmed the microcontroller to vary the coil frequency from 7 Hz to 13.5 kHz. We then recorded sound produced by the Linbot. The experimental setup is depicted in Figure 2C.

To test pulse-width-modulation (PWM) control of the Linbot, we used the microcontroller to provide a PWM signal to the coil. We then observed the effect of changing the duty cycle on the actuation scheme. We first sweep through duty cycles from 0% to 100% and then from 100% to 0%. Next, we step the Linbot between set duty cycles of 10%, 20%, 40%, and 80%. We recorded the experiments using an 18-mp Canon EOS 100D camera and an EF-S 18–55 mm f/3.5–5.6 IS STM lens. We ran this experiment once to demonstrate the capabilities of the Linbots and the output is shown in Supplementary Videos S3 and S4.

Feasibility of tactile sensing. We designed an experiment to demonstrate the tactile sensing capabilities of the Linbot by showing that the Hall-effect sensor can be used to track the movement of the magnets in three dimensions. The Z axis of the Hall-effect sensor can be used to measure force parallel to the direction of motion of a Linbot. The X and Y axes of the sensor can measure the shear force between the two halves of the Linbot. In this experiment, the top half of the Linbot is moved along the positive and negative directions of the X, Y, and Z axes. A combination of the different LEDs on the Linbot is used to display the direction of displacement. The schematic of the experiment is shown in Supplementary Figure S8. The experiment was run once to demonstrate the capabilities of the Linbots and is shown in Supplementary Video S5.

Design of the communication experiment

We designed an experiment to investigate bidirectional communication between two Linbots and to evaluate the error rate in data transmission at different communication distances. The experimental setup is shown in Figure 2E. We demonstrate communication between the Linbots using on–off keying of a 700 kHz carrier signal. The coils were not modified for demonstrating communication in any of the experiments.

In this experiment, we tested the communication between two Linbots at varying distances from one another. We increased the distance between the Linbots in steps of 5 mm from 90 to 150 mm. We measured the distance from the

centers of the Linbots. The experiment is divided into two parts, where at each distance, the first Linbot transmits a stream of data and the second Linbot receives the transmitted data, then the second Linbot transmits a stream of data and the first Linbot receives the transmitted data. The transmitted information consists of 50 bytes of data. We set these data to be the numbers 11–211 in incremental steps of 4. This array of numbers varies the shape of the signal.

The Linbot that receives the transmitted data knows the series of bytes it should be receiving, and when it receives a byte it checks the value against its expected value. If the value is different from the expected value, the Linbot interprets the byte as a faulty byte. In the experiment, the orange light depicts a Linbot in transmission mode and a green light depicts a Linbot in receiver mode. After receiving the stream of bytes, if there is a fault in the received data, the Linbot uses a combination of its LEDs to depict the number of faulty bytes received. The Linbot flashes a blue LED once for every 10 faulty bytes and flashes an orange LED once for the remaining single faulty bytes.

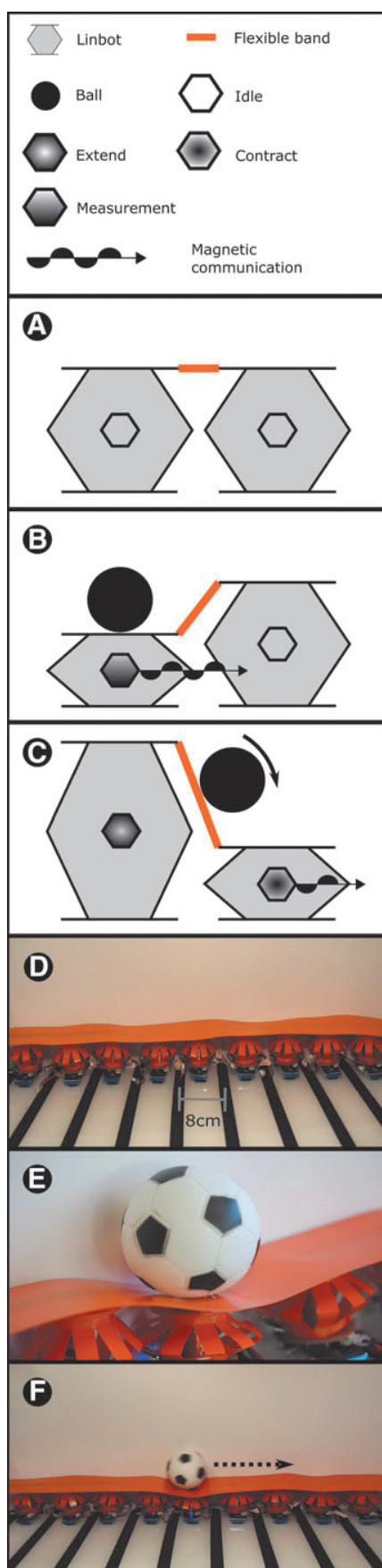
The number of wrong bytes received is used to calculate the error rate at each communication distance. If the Linbot does not receive any transmitted data because it is outside the communication range, the error rate is considered to be 1. Once a distance is reached where both Linbots have an error rate of 1, all larger distances for that pair are assumed to be 1.

Due to the large number of Linbots, we decided to perform a screening experiment by sampling more than 50% of the population, so that we could gain a good understanding of the overall capability of the Linbots. We chose five Linbots at random, from the population of nine, and used each one for two tests at each range. To separate the effects of transmission strength from the receiving sensitivity for each Linbot, we chose to test each Linbot twice, in a different pairing. By taking into account all of the permutations we calculated that we needed 500 binary tests of communication at each range. This experiment is shown in Supplementary Video S6.

Design of the Linbot collective experiments

Design of the peristaltic conveyor. In addition to the experiments that show the capabilities of one or two Linbots, we performed additional demonstrations to show how the Linbots can perform collective behaviors. For the first demonstration, we designed a peristaltic conveyor to show how a collective of our Linbots can use simple individual behaviors to perform a more complex task. The system is shown in Figure 3D. The conveyor used nine Linbots arranged in a straight line within a platform. The current used to power the Linbot communication controls the maximum range of the communication. Based on the range of around 100 mm seen in the communication test, we placed the Linbots 80 mm apart. This spacing only allowed nearest neighbor communication.

The platform consists of two parts: a backboard to prevent objects from falling off the conveyor and a base to hold the Linbots. The conveyor was given a 4° roll so that conveyed objects rest against the backboard, and a 1° pitch, so that these objects are moved uphill. For this experiment, we programmed the Linbot as a finite state machine to respond to either contact by an object or receiving a message from a neighbor. On detecting communication from another Linbot, the receiving Linbot contracts. When the Linbot contracts or



is compressed, it turns on its communication and transmits information to its neighboring Linbot. The Linbot then expands to push the object to the next Linbot. Multiple Linbots performing this sequence create a traveling wave along the conveyor from one side to the other.

After expansion, each Linbot enters an unresponsive state for a set amount of time to avoid responding to the signal from the succeeding Linbot in the conveyor. This mechanism ensures that the wave travels only in one direction, carrying the object with it. Schematics of the mechanism are shown in Figure 3A–C. We used a small toy ball that weighs 40 g for this experiment. This experiment is shown in Supplementary Video S7.

Design of the peristaltic sorter. To demonstrate how Linbots can be reconfigured to perform other tasks, we designed a peristaltic sorter using the Linbots. The layout of the Linbots for this sorter is shown in Figure 4A. In this demonstration, nine Linbots are arranged in a 3×3 array and are capable of sorting objects by their weight. A 2.5 g ball and a 40 g ball were used to demonstrate the sorting technique in this experiment. The 2.5 g ball should be moved to the left, and the 40 g ball should be moved downward.

As the sorter configuration uses multiple, nondormant Linbots within the communication range of one another, we needed to use an addressing system to control if a received command activates a Linbot. To implement the addressing system before starting the sorting task, when the Linbot collective is turned on, the bottom left Linbot is pressed. Pressing a Linbot will cause it to propagate a message through the collective, allowing each Linbot to know how far away, on the grid, it is from the Linbot pressed. The process is repeated with the top left Linbot. Based on the grid distances given by the bottom left and top left Linbot, each Linbot will have unique coordinates. Each Linbot will also know the relative position of the other coordinates on the grid. This addressing system is demonstrated in Figure 4C–E.

When a Linbot detects a weight placed on it, it increments a counter related to the class of the weight it believes is on top of it. When the counter of a class reaches a threshold, the Linbot sends a message to its neighbors causing them to actuate. This buildup and discharge behavior is inspired by action potentials in heart muscle cells.³⁴

The message transmitted by the Linbot under a weight contains the coordinates of the signaling Linbot and the direction to move the weight. The neighboring Linbots actuate based on their relative position to the instructing Linbot. This actuation leads to the ball rolling off the edge of the grid in the

FIG. 3. Peristaltic conveyor. (A) Schematic of two Linbots at rest within the peristaltic conveyor. (B) Schematic of the same Linbots within the conveyor detecting a weight placed on the first Linbot. At this point, it communicates with its neighbor. (C) Schematic of the two Linbots after the first Linbot sends the communication signal. This communication causes the second Linbot to contract, while the first Linbot extends. The actuation changes the slope of the flexible band and causes the weight to roll off and onto the second Linbot. The second Linbot then communicates to its next neighbor along the conveyor. This process is repeated to create a traveling wave along the conveyor. (D) The conveyor at rest. (E) A close-up of a ball traveling along the conveyor. (F) The conveyor moving a ball from one side to the other.

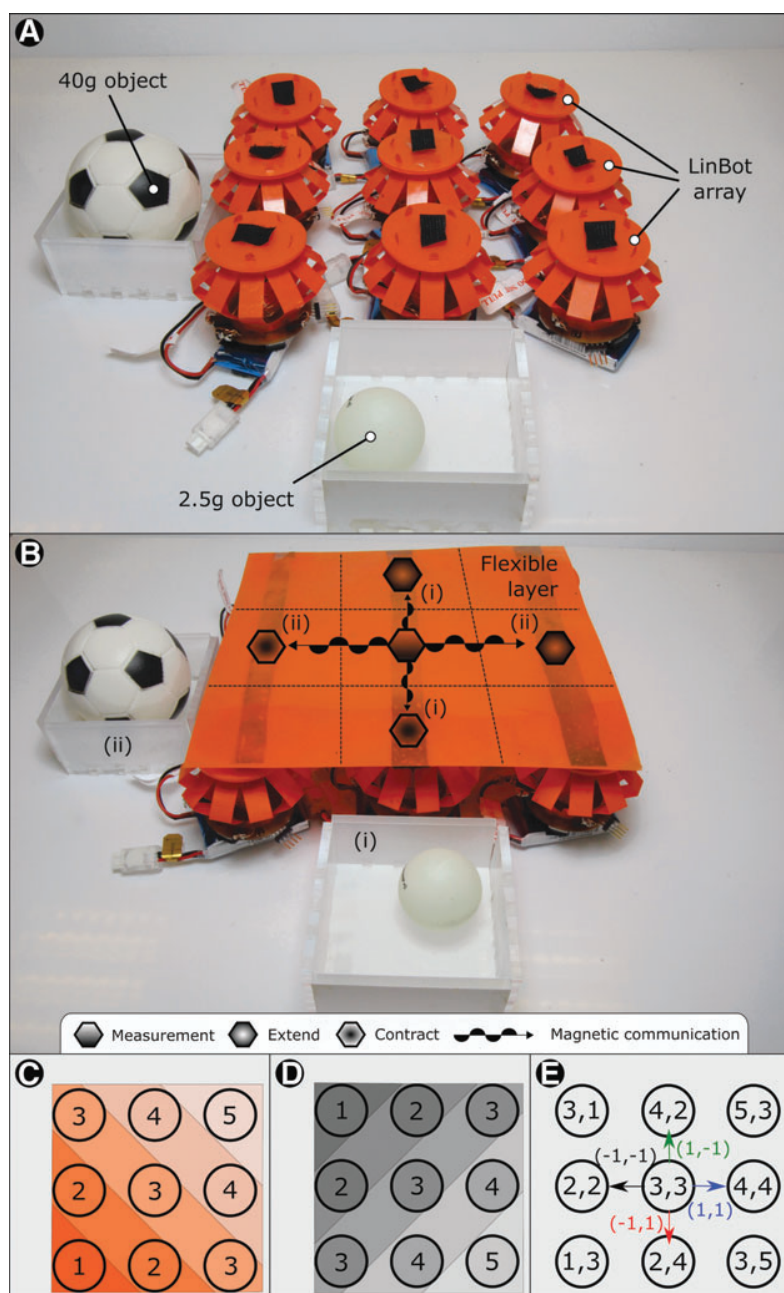


FIG. 4. Peristaltic sorter and addressing system. (A) The peristaltic sorter without the flexible layer on top of it, showing the Linbot array. (B) The peristaltic sorter with the flexible layer attached. The behavior of the sorter is shown with the central Linbot detecting the weight of an object and its neighbors actuating to roll the object in the desired direction based on weight. (i) Behavior of the sorter for the 2.5 g object. (ii) Behavior of the sorter for the 40 g object. All the Linbots in the sorting system had the same controller. (C) Shows the allocation of the first coordinate of the addressing system, based on communication hops from the bottom left Linbot. (D) Shows the allocation of the second coordinate of the addressing system, based on communication hops from the top left Linbot. (E) The final addresses for the peristaltic sorter. Vectors between neighboring Linbots in this coordinate scheme are shown around the central Linbot.

desired direction. The Linbots all have the same controller and can function in any role or position within the system. The setup and behavior of the sorter are depicted in Figure 4B. This behavior is similar to the peristaltic conveyor, detailed in the Design of the Communication Experiment section, and rolls the ball in the correct direction.

Results and Discussion

Fabrication and assembly

We designed the Linbots to have a size and weight appropriate for ease of fabrication, manufacturing, and assembly. They each have an inner diameter of 50 mm, an outer diameter of 70 mm, and a height of 50 mm at the rest position. This size

allowed us to fit all of the electronic components on one side of the PCB while also allowing large numbers of Linbots to be space efficient. Each Linbot weighs 33 g, and the battery used for each Linbot also weighs 33 g. The Linbots are designed for manufacturability and the total bill of materials is £13.64. Further information about the components and assembly can be found in The Fabrication of the Linbot section of Supplementary Data, and in the supplementary ZIP folder containing CAD files and PCB schematics.

Results of the individual Linbot characterization experiments

Quantifying force output. We expected the force output to be the sum of spring force and magnetic force between the

electromagnets and permanent magnets. When the coil and magnet are fully separated, the force is given by the following:

$$F \simeq kd + \mu \frac{q_{coil}q_{pmag}}{4\pi(d+r_0)^2}, \quad (2)$$

where F is the total force, k is the spring constant, d is the distance from rest position, μ is the permittivity of air, q_{coil} is the pole strength of the electromagnetic coil, q_{pmag} is the pole strength of the permanent magnets, and r_0 is the initial distance between center of the coil and center of the magnets. q_{coil} is given by the following:

$$q_{coil} = \frac{NIA}{L}, \quad (3)$$

where N is the number of turns, I is the current through the coil, A is the cross-sectional area of the coil, and L is the length of the coil.

When the permanent magnets enter the coil, the magnetic force between them drops, reaching zero when their centers align. At the beginning, we expect a positive force trending down linearly due to the spring force dominating. Once the centers of the permanent magnets move away from the center of the coil, we expect a rapid increase that overtakes the decrease of the spring force. Once the permanent magnets are separated from the electromagnet, further behavior will be an inverse-square relationship, summed with a negative linear term as the spring force starts pulling the magnet back.

We see this expected behavior in the results from our force test, shown in Figure 2B. One repeat of the experiment is recorded in Supplementary Video S2. The results have a standard deviation of between 0.05 and 0.21 N, and this variance is likely caused by differences during construction, leading to a spread of restoring forces in the springs for each Linbot. The springs can also exhibit some asymmetry, diverting the force from being vertically upward, leading to a reduction in measured force output.

The rest length of the Linbots used in the experiment ranged 46.5–51.0 ± 0.5 mm. Removing this variance would require using a machine to fold the springs of each Linbot. The error seen in this experiment is acceptable as these robots are soft and have not been designed to require a high level of precision to achieve their tasks; instead, they rely on compliance to adapt to the environment. The maximum force output was 0.81 N, which suggests that the Linbots could lift objects with masses below ~83 g. For objects near this limit, lifting would require a controller that actuates such that it does not allow the Linbot to compress below 0.85 relative length. Below this relative length, the force output drops off and the Linbot may be unable to lift as heavy an object.

Frequency response analysis. We expected the Linbots to actuate at a low control frequency and transition to sound output as the frequency reaches audible levels. At high frequency, we also expect to use PWM to produce forces smaller than the maximum actuation force, and therefore, partial actuation. We demonstrated that the Linbots can actuate up to high frequencies and act as a loudspeaker. The waveform for the output sound is shown in Figure 2D, and its frequency spectrum is shown in Supplementary Figure S9. We also show this behavior in Supplementary Video S3. The video shows

the Linbot actuating at increasing frequencies and then transitioning to sound output at the highest frequencies. Our use of a PWM actuation signal produced the expected partial actuation behavior. The partial actuation behavior resulting from the PWM signal is demonstrated in Supplementary Video S4.

Feasibility of tactile sensing. We designed the Linbots to be able to sense the displacement of their top half relative to their bottom half. We expected accurate classification of movement for positive and negative movements along three orthogonal axes. We demonstrated this ability in Supplementary Video S5. The video shows the Linbot being moved in six orthogonal directions, the direction of movement is displayed via the LED color combinations, shown in Supplementary Figure S8.

Evaluation of the communication between Linbots

We expected each pair of Linbots to show a very high successful transmission rate at separation distances below their maximum transmission range, as the receiving circuits have a threshold for signal strength that is set above the background noise. The background noise level from the receiver circuit is shown by Supplementary Figure S12. As they reach their maximum communication range, we expected their success rate to decrease sharply. This decrease is due to the received transmission strength approaching the threshold of the receiver, which is detailed in The Fabrication of the Linbot section of Supplementary Data. As some signals are pushed over or under the threshold by random noise, we would expect many faulty bytes. Once above the maximum transmission range, we expect none of the bytes to be received, as the transmission strength would always be below the signal threshold.

Due to variation in the construction of the coils, the transmission strength and receiving sensitivity vary in each Linbot. This variation gives each pair of Linbots a different maximum transmission range. The effect of this variation was expected to give rise to a high success region before any pairs reach their maximum range, then a distance where the average success rate is below 1 and has a high error due to some pairs having started their rapid decrease, while others have not. The average success rate should then decrease to zero as more pairs move past their maximum range.

The results of the communication test between the Linbots are shown in Figure 2F. The experiment is shown in Supplementary Video S6. We demonstrated that the Linbots can communicate with each other over at least 100 cm, giving us our expected high success region. Beyond that, the average success rate started to decline with a high error rate as expected. The shortest maximum transmission range seen was 105 mm, while the longest was over 150 mm. One Linbot did not reach a maximum transmission range. This unexpectedly high maximum range is the reason the mean error rate only reaches ~0.8.

The results of these communication experiments are used to quantify the maximum distance for reliable communications and acted as a guideline for the separation distance between Linbots in the following experiments that used collectives of robots.

Results of the Linbot collective experiments

Peristaltic conveyor. We expected a collective of Linbots to be able to generate a sufficiently powerful peristaltic wave

to roll a ball up a gradient. Our Linbot collective achieved this task and the result can be seen in Supplementary Videos S7 and S8. These videos show the Linbots rolling a ball along the conveyor by using collective behavior. We used a delay of 0.45 s before each Linbot propagated the wave, and this delay led to the ball traveling at an average of 18 cm/s across the conveyor. A close-up of the ball moving along the conveyor and the conveyor moving the ball from one side to the other is shown in Figure 3E and F, respectively.

This conveyor would be suitable for any objects that slide or roll but would have trouble moving adhesive objects. The peristaltic wave used could have been faster, as the video shows the ball sometimes getting slowed by falling into the trough of the wave. A faster wave also increases the chance that the ball would fall behind the wave and stop. The speed of the wave is limited by the gradient created for the ball to roll down. This gradient is determined by the actuation range of the Linbots and the spacing between them. In this experiment, the angle of the rising slope of the peristaltic wave is $\sim 12^\circ$, including the opposing 3° pitch of the conveyor. Increasing the range of motion of the Linbots or reducing the distance between them would allow for a faster peristaltic wave.

Two-dimensional peristaltic sorting table. We designed the Linbot collectives to be able to sense the difference between two balls, one weighing 2.5 g and the other 40 g. We also expected that they would move the 2.5 g balls off the grid to the left side and the 40 g ball off the grid to the bottom side. Our collective behaved as expected. This behavior is depicted in Figure 4B and can be seen in Supplementary Videos S9 and S10. The videos show the Linbots rolling the two balls in the correct directions.

The signal used to classify the balls is shown in Supplementary Figures S10 and S11. The difference in signal between the two balls means that comparing the signal to a threshold spaced between the two will correctly classify a ball every time. These figures also show that the rest length of a Linbot is changed even after removing the weight placed on them. In the experiment, we removed the effect of this change on the signal by resetting the base Hall-effect reading after a weight is removed. The Linbots in this sorter have identical controllers that use the relative positions of Linbots to determine actions. This distributed control means that the controller is not reliant on the size of the sorter.

Scope for Development

Vibrational sorting

The high actuation frequency of the Linbots allows them to be used for vibrational sorting. Vibrating collections of objects allow them to be sorted by firmness or density. A single Linbot with a slanted plate attached to the top can produce the vibrations needed to sort objects. The peristaltic sorter could also be adapted for a mixture of vibrational and peristaltic sorting.

Communication range

Our use of inductive data transmission for the Linbots means that communication is only possible between nearby Linbots. In the future, the frequency of the carrier wave used could be increased to create far-field communication between

Linbots and to enable long-range communication between dispersed Linbots.

Wireless charging

Through adding a rectifier and LiPo balance circuit to the Linbot, its battery could be charged through inductive power transfer. This would involve placing the Linbot in a strong alternating magnetic field, whose power would be coupled into the main coil of the Linbot.

Wireless programming

By writing a program to allow Linbots to be programmed via UART, we could allow for wireless programming of the Linbots. This modification would reduce the time needed to program the Linbots as an antenna could be used to program them simultaneously. It would also allow new programs and controllers to be propagated through a collective by the Linbots themselves.

Additional sensors

By connecting I²C sensors to Linbot PCB I²C bus pins, we can tailor the Linbots to new tasks. For example, to make an array of Linbots respond to the distance from objects, IR range sensors can be added to a Linbot PCB.

Learned behavior

In large Linbot collectives, each individual robot can affect the state of other Linbots with whom it does not have a direct communication link. This complication leads to difficulties in writing effective controllers. Machine learning algorithms could be applied to allow Linbots to learn the dynamics of their collective and even to have individually tailored controllers.

Self-synchronization

If we have a task that requires synchronization, we can design our Linbot controllers to create a global clock. Using phase rate equalization, any detectable pulses can be used to synchronize the Linbots.³⁵ We can create these pulses with periodic broadcasts from a Linbot to neighboring Linbots. We can also use the periodic actuation of a Linbot, which can detect the actuation of neighboring Linbots through tactile signals from a flexible layer on top of them.

A larger peristaltic sorter

Further robots could be added to make the peristaltic sorter larger without needing to rewrite the controller. This larger sorter could be used to examine large-scale behaviors of Linbots.

Locomoting Linbot collectives

A Linbot collective could be reconfigured for locomotion, similar to the Wormbot.¹⁶ The peristaltic conveyor could also be inverted to produce waves that would enable the whole assembly to locomote across a surface. Reducing the weight of the Linbots by decreasing the battery size would allow for more robust locomotion. A 2D Linbot collective designed for

locomotion would be capable of steering in response to sensory information.

Miniaturized Linbots

The Linbots could be miniaturized for use in different applications such as tactile displays or implantable devices. By converting the circuitry into one integrated circuit, the volume of a Linbot could be reduced to about a cubic centimeter. At this size, the Linbots could use the magnets and coils from the smallest commercially available solenoids, while still using off-the-shelf batteries.

Making the robot even smaller would preclude the use of available batteries. This means that either a miniature battery would need to be manufactured or the robot would need to run on miniature charged capacitors and an external power source. At smaller scales, the coil would need to be changed from being a wound wire to a planar coil cut into a PCB, and the magnets would need to be fabricated rather than bought. The limit at this level would be the minimum size of integrated circuits, which means the Linbot cannot be smaller than a few millimeters in size.

Conclusions

Modular robotic collectives are more robust than monolithic systems. Often modular systems are limited in their functionality due to cost. Soft, modular robots also often require an outside pressure or vacuum source. We present our Linbots, our combination of untethered, reconfigurable robots with soft robotics. Our Linbots demonstrate communication, actuation, tactile sensing, proprioception, and sound synthesis. The linear motion of our Linbots sets them apart from existing modular robots and allows them to be used for tasks that require peristaltic motion.

We demonstrated the abilities of individual Linbots as well as their ability to communicate with one another. We used a Linbot collective to convey a ball up a slope using a peristaltic wave. We then reconfigured this Linbot collective to demonstrate sorting. The collective was able to sort two balls based on weight and transfer them to desired bins.

Our use of multifunctional hardware allows the system to be low cost without sacrificing functionality. The soft nature of our Linbots allows them to conform to their surroundings. Our Linbots provide a low-cost modular platform that can be configured for different real-world tasks. In addition, our multifunctional voice coil system can be adapted to other modular or swarm robotic systems. Our work supports the move toward reconfigurable, modular robotic platforms as useful tools for both academia and industry.

Funding

This study was supported by EPSRC via the Robotarium Capital Equipment and CDT Capital Equipment Grants (EP/L016834/1), and the CDT in Robotics and Autonomous Systems. M.P.N. gratefully acknowledges support from the CDT in Intelligent Sensing and Measurement (EP/L016753/1), United Kingdom. A.A.S. acknowledges support from the EPSRC ORCA Hub (EP/R026173/1).

Authors' Contribution

R.M.M., designing and developing the Linbot, developing the experimental scheme, writing and revising the article.

M.E.S., designing and developing the Linbot, developing the experimental scheme, writing and revising the article. M.P.N., writing parts of the article/developing the figures, revising the article, and contributing to Linbot PCB design. B.W.F., advised on building the Linbot system. A.A.S., lead advisor and primary editor of the article.

Author Disclosure Statement

No competing financial interests exist.

References

1. Yim M, Shen W, Salemi B, *et al.* Modular self-reconfigurable robot systems [Grand Challenges of Robotics]. *IEEE Robot Autom Mag* 2007;14:43–52.
2. Clune J, Mouret J-B, Lipson H. The evolutionary origins of modularity. *Proc R Soc Lond B Biol Sci* 2013;280:1755.
3. Belke CH, Paik J, Mori A. A Modular Origami Robot, *IEEE ASME Trans Mechatronics* 2017;22:2153–2164.
4. Neubert J, Rost A, Lipson H. Self-soldering connectors for modular robots. *IEEE Trans Robot* 2014;30:1344–1357.
5. Stoy K. Reconfigurable robots, in *Springer Handbook of Computational Intelligence*, Kacprzyk J, Pedrycz W. (Eds). Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 1407–1421.
6. Mahon ST, Roberts JO, Sayed ME, *et al.* Capability by stacking: The current design heuristic for soft robots. *Bio-mimetics* 2018;3:16.
7. Sproewitz A, Billard A, Dillenbourg P, *et al.* Roombots-mechanical design of self-reconfiguring modular robots for adaptive furniture. In *ICRA*, Kobe, Japan, 2009:4259–4264.
8. Vergara A, Lau Y-S, Mendoza-Garcia R-F, *et al.* Soft modular robotic cubes: Toward replicating morphogenetic movements of the embryo. *PLoS One* 2017;12:1–17.
9. Kim S-R, Lee D-Y, Koh J-S, *et al.* Fast, compact, and lightweight shape-shifting system composed of distributed self-folding origami modules. In *ICRA*, Stockholm, Sweden, 2016:4969–4974.
10. Stanley AA, Okamura AM. Controllable surface haptics via particle jamming and pneumatics. *IEEE Trans Haptics* 2015;8:20–30.
11. Follmer S, Leithinger D, Olwal A, *et al.* inFORM: dynamic physical affordances and constraints through shape and object actuation. *UIST*, St. Andrews, UK, 2013:417–426.
12. Rubenstein M, Ahler C, Nagpal R. Kilobot: A low cost scalable robot system for collective behaviors. In *ICRA*, St. Paul, MN, 2012:3293–3298.
13. Skjetne C, Haddow PC, Rye A, *et al.* The ChIRP robot: A versatile swarm robot platform. *RiTA*, Denver, CO, 2013: 71–82.
14. Nemitz MP, Sayed ME, Mamish J, *et al.* HoverBots: Precise locomotion using robots that are designed for manufacturability. *Front Robot AI* 2017 Nov;4(55):00055.
15. Nemitz MP, Marcotte RJ, Sayed ME, *et al.* Multi-functional sensing for swarm robots using time sequence classification: HoverBot, an example. *Front Robot AI* 2018;5:55.
16. Nemitz MP, Pavel M, Barraclough TW, *et al.* Using voice coils to actuate modular soft robots: Wormbot, an example. *Soft Robot* 2016;3:198–204.
17. Stokes A, Shepherd R, Morin S, *et al.* A hybrid combining hard and soft robots. *Soft Robot* 2014;1:70–74.
18. Kim S, Knoll T, Scholz O. Feasibility of inductive communication between millimeter-sized wireless robots. *IEEE Trans Robot* 2007;23:605–609.

19. Paulino T, Ribeiro PC, Neto M, *et al.* Low-cost 3-axis soft tactile sensors for the human-friendly robot Vizzy. In ICRA, Singapore, 2017:966–971.
20. Scott W, Stevens J, Binder-Macleod SA. Human skeletal muscle fiber type classifications. *Phys Ther* 2001;81:1810–1816.
21. Robertson MA, Paik J. New soft robots really suck: Vacuum-powered systems empower diverse capabilities. *Sci Robot* 2017;2:eaan6357.
22. Acome E, Mitchell SK, Morrissey TG, *et al.* Hydraulically amplified self-healing electrostatic actuators with muscle-like performance. *Science* 2018;359:61–65.
23. Kellaris N, Venkata VG, Smith GM, *et al.* Peano-{HASEL} actuators: Muscle-mimetic, electrohydraulic transducers that linearly contract on activation. *Sci Robot* 2018;3:eaar3276.
24. Angatkina O, Chien B, Pagano A, *et al.* A metameric crawling robot enabled by Origami and smart materials. In ASME 2017 Conference on Smart Materials, Adaptive Structures and Intelligent Systems, Snowbird, UT, 2017: 3836–3846.
25. Wei T, Stokes A, Webb B. A soft pneumatic Maggot robot. In Biomimetic and Biohybrid Systems. Cham, 2016:375–386.
26. Menciassi A, Gorini S, Pernorio G, *et al.* A SMA actuated artificial earthworm. In ICRA, New Orleans, LA, 2004: 3282–3287.
27. Fukuda T, Hosokai H, Otsuka M. Autonomous pipeline inspection and maintenance robot with inch worm mobile mechanism. In ICRA, Raleigh, NC, 1987:539–544.
28. Boxerbaum AS, Shaw KM, Chiel HJ, *et al.* Continuous wave peristaltic motion in a robot. *Int J Robot Res* 2012;31: 302–318.
29. Stommel M, Xu WL, Lim PPK, *et al.* Soft peristaltic actuation for the harvesting of Ovine Offal. *Robot Intelligence Technology and Applications 3: Results from the 3rd International Conference on Robot Intelligence Technology and Applications* Kim J, Yang W, Jo J, Sincak P, Myung H. (Eds). Cham: Springer International Publishing, 2015, pp. 605–615.
30. Stommel M, Xu W. Optimal, efficient sequential control of a soft-bodied, peristaltic sorting table. *IEEE Trans Autom Sci Eng* 2016;13:858–867.
31. Sung C, Rus D. Automated fabrication of foldable robots using thick materials. In *Robotics Research: Volume 1* Bicchi A, Burgard W. (Eds). Cham: Springer International Publishing, 2018, pp. 253–266.
32. STMicroelectronics, Home Page. [Online]. Available: www.st.com/content/st_com/en.html. Accessed February 11, 2018.
33. Osborne A. *An Introduction to Microcomputers: Basic Concepts*, first ed., Osborne/McGraw-Hill, 1980.
34. Cheney N, Clune J, Lipson H. Evolved electrophysiological soft robots. *ALIFE* 2014;14:222–229.
35. Brandner G, Schilcher U, Bettstetter C. Firefly synchronization with phase rate equalization and its experimental analysis in wireless systems. *Comp Netw* 2016;97:74–87.

Address correspondence to:

Adam A. Stokes

Scottish Microelectronics Centre

School of Engineering

Institute for Integrated Micro and Nano Systems

The University of Edinburgh

Alexander Crum Brown Road, King's Buildings

Edinburgh EH9 3FF

United Kingdom

E-mail: adam.stokes@ed.ac.uk

3.3 Supplemental Videos

For the supplemental videos for this publication scan this QR code:



3.4 Conclusion

In order to build a modular robotic sorting table I built a novel modular robot called the Linbot based on a previous Wormbot design. My Linbots are the first modular, robotic peristaltic cells. I have shown the manufacturability of my Linbots through constructing a small collective of nine Linbots.

I tested the abilities of individual units in order to understand the limitations when it came to assembling Linbot collectives and the tasks that they could achieve. While the characteristics of the Linbots were not identical between units, they all displayed an adequate minimum of abilities in terms of sensing, communication and actuation which meant that Linbot collectives will be capable of low weight (under 100g per object) sorting tasks. The frequency response experiment shows that the Linbots can produce high frequency actuations that could be used for vibrational sorting.

I used my Linbot collective to show that combined Linbots provide more useful work than the individual robots could achieve alone. Through individual computation, actuation and communication my Linbot collective acted as a conveyor and it could be reconfigured into a sorter using the same Linbots.

Chapter 4

Simulating Peristaltic Tables and Optimising Peristaltic Waves

4.1 Introduction

The high dimensionality of multi-robot systems can make it difficult to write optimal controllers. Creating a simulator for a peristaltic table allowed me to quickly test new controllers. My simulator also allowed me to quickly optimise machine learning based controllers. My new simulator can also be used to simulate other peristaltic tables and so I have made it available as the python package PeriSim for use in others' work.

4.1.1 Making a New Simulator

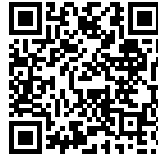
I decided to avoid the use of existing simulators such as Gazebo [52] or Unity [53] as they do not have a developed simulation of soft materials such as the surface of a peristaltic table. These simulations can be handled by a Finite Element Analysis (FEA) simulator such as SOFA [54]. FEA simulators are very high fidelity but require a large amount of computational power to run, making machine learning in those simulators difficult. I therefore decided to make my own simulation that handled the soft surface of the table as a series of overlapping Gaussians. This simulator modelled the key factors involved in moving objects across a peristaltic table while still being computationally cheap to run. To reduce the reality gap caused by the unrealistic behaviour inherent in such a simple simulation I used a radical envelope of noise. The radical envelope of noise is a technique that involves varying the physical parameters of the simulation, like the strength of gravity, between each simulation run. Controllers optimised in this simulation should be robust to these changes in physical parameters and should be more robust to the change from simulated to real-world environments. For more information see section 2.4.3.

4.1.2 PyTorch

Tensor packages presented an opportunity to parallelise the calculations through matrix operations. I decided to use PyTorch [55] as it offered a good mix between functionality and ease of use. Using PyTorch also allows the use of GPU acceleration for the calculations although my tests suggest that this doesn't provide a speedup until the simulated peristaltic table contains thousands of cells.

4.1.3 PeriSim

I developed the simulation and used it for a proof of concept optimisation of a peristaltic wave. I then submitted this work to the journal *Advanced Intelligent Systems* where it has been published. To view the source code and instructions for PeriSim scan the QR code below:



4.2 PeriSim Publication

PeriSim: A Simulator for Optimizing Peristaltic Table Control

Ross M. McKenzie, Jamie O. Roberts, Mohammed E. Sayed, and Adam A. Stokes*

Peristaltic conveyance can be used for the sorting and transport of delicate and nonrigid objects such as meat or soft fruit. The non-linearity and stochastic behavior of peristaltic systems make them difficult to control. Optimizing controllers using machine learning represents a promising path to effective peristaltic control but currently, there is no suitable simulated model of a peristaltic table in which to run these optimizations. A simple, simulated model of a peristaltic conveyor that can be used for optimizing peristaltic control on a variety of peristaltic tables is presented. This simulator is demonstrated through a limited control problem evaluated on our real-world system that is built for peristaltic conveyance. This simulator is available as the python package PeriSim so that it can be used by the robotics community for peristaltic control development.

been created and used for simple tasks.^[2–6] The interest in peristaltic motion comes from the limitations of existing 2D conveyance systems seen in the industry. Industrial 2D conveyance relies on rollers or wheels that push against rigid, flat surfaces^[7,8] and is not suitable for soft, slippery, or round objects. These objects can be manipulated with peristaltic motion, allowing for the automation of new areas such as offal harvesting.^[1]

A major barrier to the adoption of peristaltic systems in industry is in their control. Peristaltic waves are challenging to control because they are nonlinear, temporal combinations of the movements of multiple cells within the table.^[9] The effect of the waves on objects is also highly dependent

on the physical characteristics, velocity, and orientation of the objects. The non-linearity of the surface to object interactions makes it difficult to accurately predict the precise effect that a passing peristaltic wave will have on an object.

1. Introduction

1.1. Peristaltic Motion Can be Used to Sort Delicate and Soft Objects

Peristaltic tables consist of a number of shape-changing cells that connect to form a deformable 3D surface.^[1] These cells are actuated linearly to deform the surface allowing for the automated control of gradients across the surface, as shown in **Figure 1**. Objects that slide or roll can be moved by keeping the gradient at the position of the object biased in the direction of desired travel. This control is usually achieved through a wave-like pattern of cell actuation called a peristaltic wave. A number of hardware implementations of peristaltic systems have already


1.2. Peristaltic Control Can be Optimized in Simulation

Stommel and Xu presented a probabilistic automaton for high-level peristaltic control.^[9,10] Their peristaltic probabilistic automaton uses input from the sensors of the table to determine a state and has a number of high-level wave patterns as actions. The probabilistic result of each state-action pair is determined through multiple trial runs of the system. This approach is independent of the hardware of the system and the object being manipulated, which makes it widely applicable. This approach requires a simulated model of the system as the state-action space can be very large and can take a prohibitive amount of time to explore.

The probabilistic automata approach requires that high-level actions are determined in advance to reduce the size of the state-action space. These high-level actions can be seen as a subset of the larger problem of peristaltic control as they have the same challenges in modeling caused by the nonlinear nature of peristaltic motion. Optimizing these actions provides a foundation for higher-level control and can provide valuable lessons for optimization at this higher level.

Probabilistic automaton control has been demonstrated on an inverse caterpillar motion table by Deng et al.^[11] This table moves an object by lifting it with deformable cells and then moving the point of contact between the cell and the object in the plane of the table. As this type of motion does not rely on objects rolling or sliding, simulations of this table will be mechanically simpler and less stochastic than a simulation of peristaltic motion.

R. M. McKenzie, J. O. Roberts, Dr. M. E. Sayed, Dr. A. A. Stokes
School of Engineering
Institute for Integrated Micro and Nano Systems
Scottish Microelectronics Centre
The University of Edinburgh
Alexander Crum Brown Road, King's Buildings, Edinburgh EH9 3FF, UK
E-mail: adam.stokes@ed.ac.uk
R. M. McKenzie, J. O. Roberts
EPSRC CDT in Robotics and Autonomous Systems
Edinburgh Centre for Robotics
Edinburgh, UK

 The ORCID identification number(s) for the author(s) of this article can be found under <https://doi.org/10.1002/aisy.201900070>.

© 2019 The Authors. Published by WILEY-VCH Verlag GmbH & Co. KGaA, Weinheim. This is an open access article under the terms of the Creative Commons Attribution License, which permits use, distribution and reproduction in any medium, provided the original work is properly cited.

DOI: 10.1002/aisy.201900070

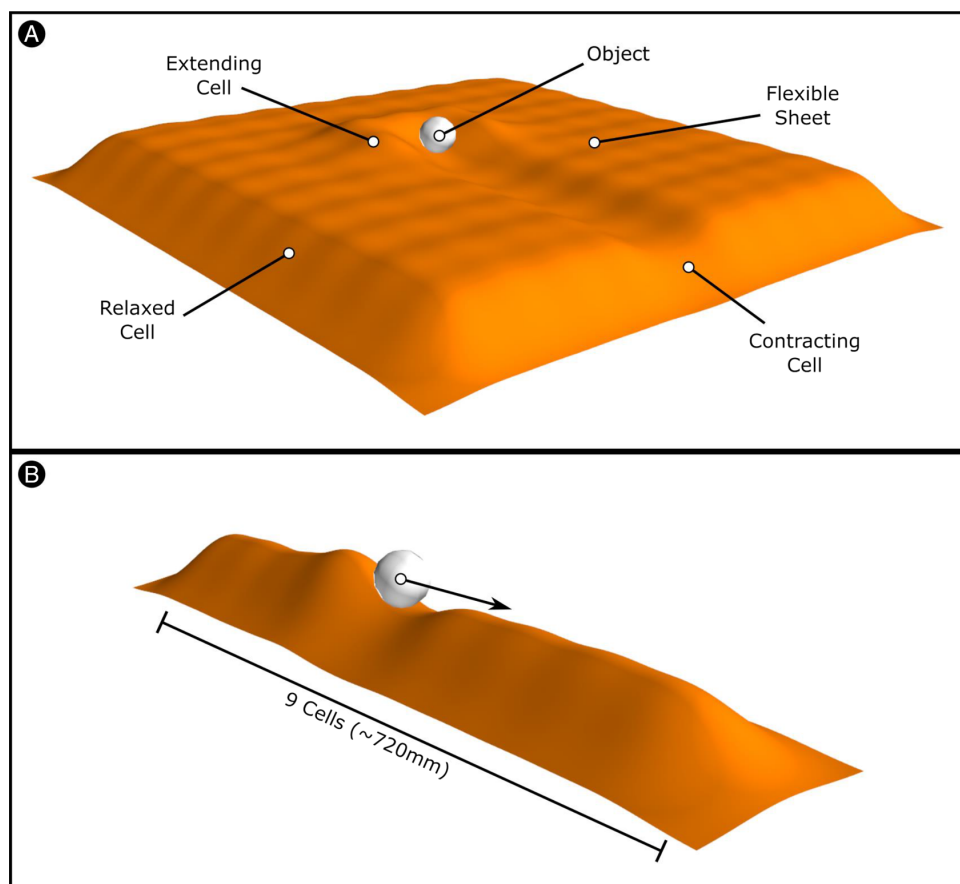


Figure 1. Simulated peristaltic tables. a) A 2D peristaltic table capable of moving an object in the plane of the table. b) The simulated 1D peristaltic conveyor used to optimize peristaltic waves for conveying the object from one end of the conveyor to the other in as short a time as possible.

1.3. Genetic Algorithms Are Ideal for Optimizing Peristaltic Waves

A basic high-level action for a peristaltic table is a single, constant amplitude wave. These waves are created through a series of cells executing a cyclic motion with a delay between their start times. This delay can be the same, for a constant velocity wave, or different to have a wave with acceleration. The exact series of delays to best move an object is highly dependent on the object and peristaltic table hardware. Genetic algorithms have been used successfully for optimizing both high-level^[12] and low-level^[13] robotic behavior. Cheney et al. demonstrated that genetic algorithms can successfully harness soft material properties to produce effective behavior.^[14] This work suggests that a genetic algorithm approach could work for optimizing peristaltic control due to the shared theme of independent, nonrigid components operating together to produce useful work. Genetic algorithms present a simple optimization method to test the feasibility of optimized peristaltic control. This optimization will require a simulated peristaltic table to explore the state space in a reasonable time. This simulation would be imperfect as the nonlinear nature of the system that creates the need for probabilistic automaton control also makes modeling the system in simulation difficult.

1.4. Radical Envelope of Noise to Overcome the Reality Gap

Optimizations will often exploit aspects of a simulation that are different or missing in the real world.^[15] Examples of aspects that often change when moved to the real world include minor variance in actions that are perfectly repeatable in simulation, the frictional behavior of sliding surfaces changing from the simulated model, or internal computation, taking time rather than being “instant” as in the simulation. The lack of, or difference between, these elements can cause reduced performance when that system is used in the real world. This effect is called “the reality gap.” The reality gap can be reduced by learning an offset from simulation to the real world.^[16,17] It is also possible to design a system in such a way that a small section of it can be trained in the real world to overcome the differences.^[18] These methods offer a more directed approach to close the reality gap; however, they require major modifications to the genetic algorithm approach.

The reality gap can also be narrowed through altering the simulation itself. Adding noise to sensor data and uncertainty to the results of actions in a simulation makes controllers that are optimized in that simulation more robust in the real world.^[15] However, choosing the correct level of noise is

necessary for this approach and it still requires a well-validated simulation to work.

An improvement on noisy simulations is to design a simulator around the radical envelope-of-noise hypothesis.^[19] This hypothesis requires that each simulation parameter or aspect is randomly varied between each simulation run. By randomly changing the parameters that control the dynamics of our simulations between runs, we can create controllers that are robust to changes in the underlying behavior of their reality. These controllers can, therefore, be optimized in a simplistic simulator and still work in the real world. The simulator we use must still model all of the key features of the task and system we optimize but can be less complex and rely on less real-world validation.

2. Experimental Section

2.1. Simulation of a Peristaltic Table

2.1.1. Modeling Peristaltic Cells and Objects on the Table

We modeled a peristaltic table as a flexible and extensible surface in the horizontal (XY) plane that was vertically (Z) perturbed by peristaltic cells. Each peristaltic cell was modeled as a Gaussian disturbance on the surface. The amplitude of the Gaussian was determined by the height of the cell. The effect that a peristaltic cell (p) has on the height (h) of the surface at the position of each piece of cargo is

$$h_p = Ae^{-\frac{(\vec{c}-\vec{b}_p)^2}{2\sigma^2}} \quad (1)$$

where A is the height of the peristaltic cell, \vec{b}_p is the XY position vector of the peristaltic cell, and \vec{c} is the XY position vector of the cargo. The term σ is a constant controlling the width of the effect from each peristaltic cell and is the same for all cells. We chose to refer to σ as the standard deviation as it contributes to the shape of the Gaussian disturbance of the cell in the same way that the standard deviation contributes to the shape of a normal probability distribution.

From Equation (1), the effect that each of the peristaltic cells has on the gradient at the position of an object on the surface is

$$\frac{dh_p}{d\vec{c}} = -\frac{(\vec{c}-\vec{b}_p)}{\sigma^2} Ae^{-\frac{(\vec{c}-\vec{b}_p)^2}{2\sigma^2}} \quad (2)$$

The total gradient at the position of an object is a sum over all of the peristaltic cells.

The magnitude of the vertical force applied between the cargo object and the surface (F_v) is

$$F_v = -(mg + F_{act}) \quad (3)$$

where m is the mass of the object, g is the strength of gravity in the simulator, and F_{act} is the magnitude of the force produced by actuating peristaltic cells beneath the object. The value of F_{act} is 0 in all cases other than when the nearest cell to the object actuated within the last t_{act} seconds. The term t_{act} represents the duration of a movement between the actuation states of a cell and is a constant.

The vector angle between the gradient of the surface and the horizontal in the XZ and YZ planes ($\vec{\theta}$) can be calculated as

$$\vec{\theta} = \tan^{-1} \left(\sum_p \frac{dh_p}{d\vec{c}} \right) \quad (4)$$

Geometrically, the angle between the vertical force and the line normal to the plane of the surface is equal to $\vec{\theta}$, shown in **Figure 2**. The magnitude of the reaction force of the surface on the object in XZ and YZ planes (\vec{F}_r) is equal and opposite to the component of the vertical force acting normal to the surface in those planes. Therefore, using Equation (1),

$$\vec{F}_r = (mg + F_{act}) \cos(\vec{\theta}) \quad (5)$$

Geometrically, the angle between the reaction force and vertical is also equal to $\vec{\theta}$, shown in Figure 2. The acceleration of the object in the horizontal plane (\vec{a}) can be calculated in an element-wise manner using

$$\vec{a}_i = \frac{\vec{F}_r \sin(\vec{\theta}_i)}{m} - D\vec{v}_i \quad (6)$$

where D is the constant drag factor and \vec{v} is the velocity of the object.

Each simulation step proceeds in the following order: For each object, 1) calculate the gradient at the position of the object; 2) calculate the acceleration; 3) add one timestep of acceleration to the velocity; 4) add one timestep of velocity to the position.

The objects in our simulator were modeled as point masses and as such, their physical volumes did not interact with the surface or with other objects. This behavior is suitable for modeling other peristaltic tables as from our research into peristaltic table development (see Section 1.1); most systems do not rely on

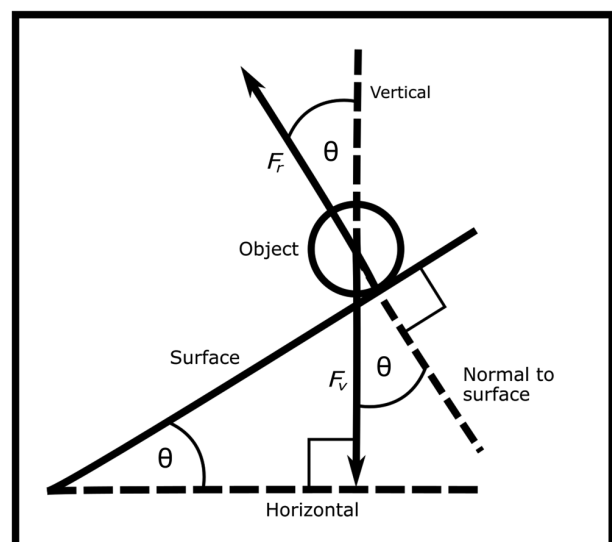


Figure 2. Schematic of an object on the surface of a peristaltic table. The term θ is the angle between the gradient of the surface and horizontal, F_v is the vertical force applied between the cargo object and the surface, and F_r is the reaction force of the surface on the object.

object-surface interactions other than moving along gradients and do not intentionally bring multiple objects into contact with each other. Our simulator used PyTorch tensors for most of the calculations it performed. PyTorch is a deep-learning python library that provides access to high-performance tensor operations.^[20] The PyTorch tensors we used in our simulator could contain information about all the objects or cells on the table, allowing calculations involving every cell and every object to be completed with one-tensor operation without looping over variables.

While the default values of the parameters of the simulator were tuned to model the hardware we used for testing, the parameters were easy to access and change. Other peristaltic table researchers can tune these parameters to model their own systems in our simulator without needing to change the source code.

2.1.2. Radical Envelope of Noise

We implemented the radical envelope of noise in this simulator by a perturbing every base parameter randomly at the beginning of each simulation run. For simulation run (i), a parameter (ω_i) has a default value (ω_0) and a range around that default value in which it may lie in each simulation. We referred to this range as the noise level (N). At the start of each simulation run, each parameter was calculated such that

$$\omega_i = \omega_0 + rN\omega_0 \quad (7)$$

where r is a random number picked from a uniform distribution between -1 and 1 . The noise level therefore could have a significant effect on the behavior of the simulation created at each run.

2.2. Design of the Peristaltic Conveyor

To test our simulation, we designed a peristaltic conveyor based on Linbot actuators and in a similar design to our previously demonstrated peristaltic conveyor.^[6] Linbots are modular robots that function as the cells in a peristaltic system. They are capable of actuating linearly, tactile sensing, and communicating with one another. To test our simulator, we arranged nine Linbots in a row to make a one dimensional peristaltic table, or peristaltic conveyor. We enclosed the conveyor with walls on both sides to stop objects from falling off. We used a sheet of acetate on top of the cells for the surface of our peristaltic conveyor. The experimental setup is detailed in **Figure 3**. We then created a model of the conveyor in our simulator with a 9×1 array of peristaltic cells and tuned the parameters of the cells to match the behavior of Linbots. We used the same control architecture for the simulated model and real-world system. This architecture is detailed in Section 2.3.1

2.2.1. Simulator Calibration with Real-World Data

We set the parameters of the simulator to match our real-world system. As the simulator is designed to be simplistic, it uses parameters that do not have direct real-world versions or that cannot be easily measured, such as the standard deviation of the Gaussians modeling the peristaltic cells. To set these values, we compared the behavior of simulation runs against the real-world experiments.

We ran our real-world system with an identical known delay for each Linbot and filmed the results. We used a circle detection algorithm from OpenCV^[21] to track the position and velocity of

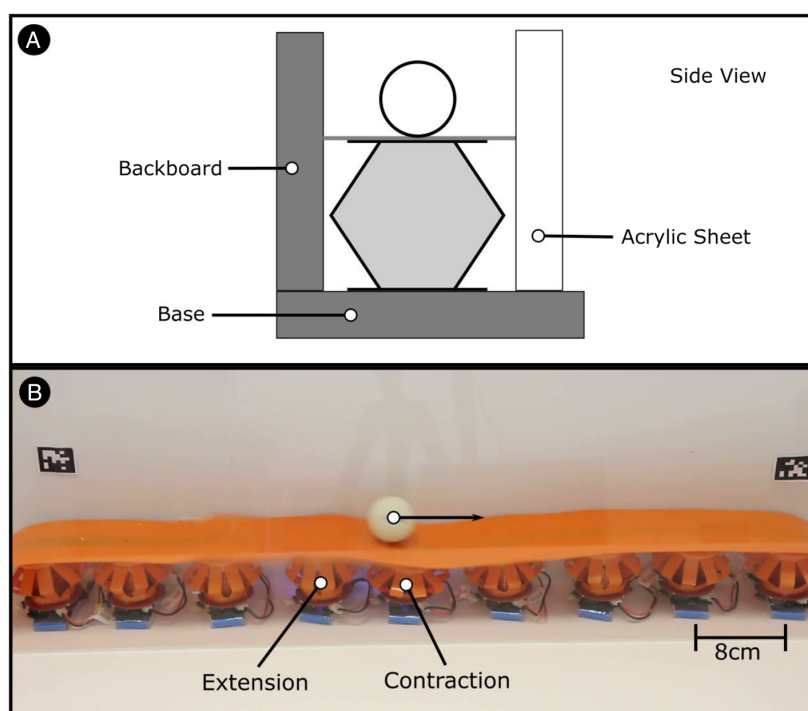


Figure 3. System overview. A) A side view schematic of the experimental setup. B) The peristaltic conveyor during an experiment.

the ball and compared the results to those of the same experiment in the simulator. We then calibrated the parameters of the simulator to produce the most realistic results possible. The simulator runs still showed unrealistic effects caused by the simplicity of the simulator, but the results were close enough to allow us to optimize robust controllers.

2.3. Design of the Control

2.3.1. Genetic Controller

We designed a state machine controller that has an identical structure for each peristaltic cell but each has an individual

control variable. The nine control variables were stored in a genetic code that is used to define a collective wide controller. Each variable was between zero and one. Each variable described the time, in seconds, that the respective peristaltic cell will remain in a contracted or extended state. The position of a gene describing the delay of a peristaltic cell in the genetic code was determined by the position of the corresponding cell in the conveyor. The state machine for each cell and the related extensions are shown in **Figure 4**. A simulated model using this controller to create a constant velocity peristaltic wave is shown in Video S1, Supporting Information. The control variables used in Video S1, Supporting Information, are all equal to 0.3.

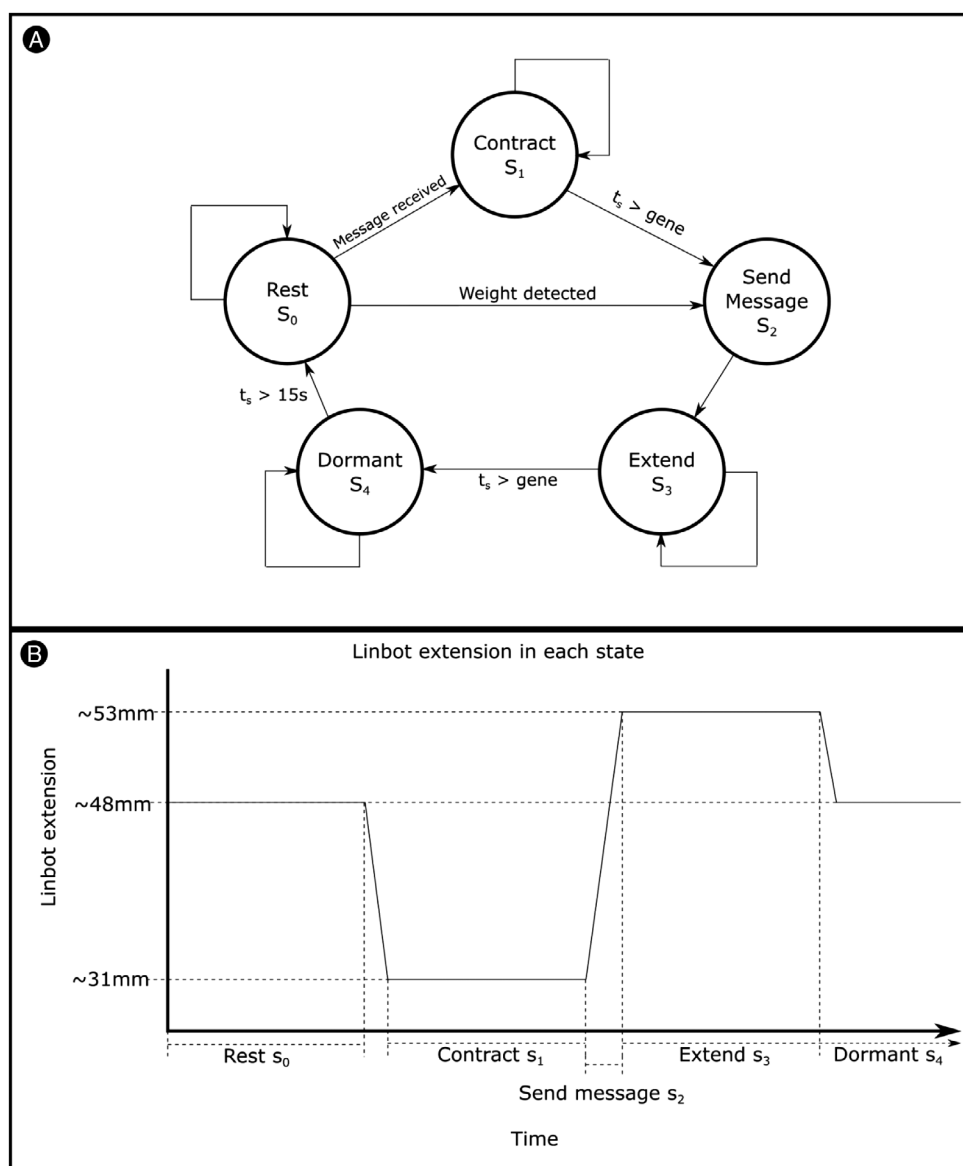


Figure 4. System overview. A) A schematic showing the state machine controller for each peristaltic cell with delays related to the genetic code. Unlabeled links represent “else” actions. B) The extension of the peristaltic cell in each state.

2.3.2. Optimizing the Controller

We generated a population of random genetic codes to use as controllers for the simulated conveyor. Each time a controller was run a fitness value was calculated and stored. During the optimization process, the fitness of each controller was averaged over ten runs in a simulation.

The fitness (f) is calculated using

$$f = -x_{\min} - 2t \quad (8)$$

where t is the time taken for the task to finish with a maximum of 10 s, and x_{\min} is the minimum distance to the end of the conveyor reached during the test; these variables are clarified in Figure 5A. We doubled t to increase evolutionary competition in more successful controllers, see below.

We optimized the fitness of the controllers via a genetic algorithm approach in simulation. We used a genetic algorithm closely aligned with previous approaches,^[22] which is demonstrated in Figure 5B. A population of controllers was randomly created and tested. We then created a new population based on members of the previous population. The probability of a controller being selected to be used for the creation of a member of the next generation is proportional to the relative fitness of the controller compared with the rest of the population. Once chosen, the selected controller underwent a crossover with a different controller selected with the same rule. The two resulting controllers then had a small chance for each gene to be mutated to create a new genetic code used for a controller in the next generation. Crossovers split the codes of two parent controllers and then created a child code by combining the first section of one parent code with the second of another and vice versa for a second child code. Mutations involved changing the value of one or more randomly selected genes in a child code within a small envelope (<0.1).

During testing, we discovered that controllers that only moved the object a short distance along the conveyor had large x_{\min} values compared to the maximum value of t . This large difference led to more successful controllers having small differences in relative fitness when compared to the minimum fitness value of the population. This narrowing of comparative fitness values reduced the selection pressure on successful controllers and reduced the exploitation of different successful strategies. We rectified this imbalance by doubling t in our fitness calculations.

For our optimization, we used 50 generations with a population of 1000 per generation.

2.4. Sampling from Noise Levels

Building higher levels of noise into the parameters of a simulator led to the optimization of controllers to work for a wider range of different parameters, making them more likely to work when moved to real-world robotic system. These controllers, however, could not exploit their environment to complete a task as much as if lower noise levels were used in the simulator. This trade-off means that controllers from a variety of noise levels need to be sampled to find the best controller. These noise levels were a percentage by which each base parameter could be perturbed at each simulation run and that the acceleration of the object could be

perturbed in each simulation timestep. We varied the noise levels from 0% to 50% in steps of 5%.

2.5. Design of the Physical Experiments to Evaluate the Optimized Controllers

To determine the best real-world controller, we tested the best controllers from each optimization. We selected the top five controllers from each optimization and programmed the control variables into our peristaltic conveyor. We then tested each selected controller by placing a ball on the right-hand half of the first cell and recording the resulting peristaltic conveyance. We calculated the fitness of each controller using Equation (7). We repeated the test ten times for each selected controller and averaged the measured fitness values.

The data collected during the experiment was in the form of videos and so we needed to extract the necessary features using computer vision. We developed a tool to evaluate the peristaltic controllers using Python with OpenCV and AprilTag libraries.^[23] The AprilTags served as the static and reliable start and finish positions with which to evaluate each peristaltic conveyance. We placed the AprilTags above the first and last cells of the system. AprilTag detection is robust, reliable, and is handled by the supporting libraries. We used the OpenCV implementation of the Hough Circle detector^[24] to detect the ball in each frame. We tuned the Hough Circle detector parameters for each video via an interactive script that saved manual changes in parameters to file. We later used the parameters in an automated script for controller evaluation. We developed our evaluation tool to be able to detect when a run is finished, classify whether the run is a success, and record the time and distance information from each run. A screenshot of our evaluation tool is shown in Figure 5A.

3. Results and Discussion

3.1. Using PeriSim

We made our simulator available as a python package called PeriSim, which can be installed via `pip install perisim`. Full instructions and the source code are available at: <https://github.com/stokesresearchgroup/perisim>. PeriSim provides a python object that stores the state of a peristaltic table and calculates the next state. The user provides variables for the initial state of the system and controls the system via writing a controller object or by using the PeriSim object functions to control the cells and updates. PeriSim can also create 3D visualizations of the simulator state via `mlab`,^[25] examples are shown in Figure 1 `mlab` is a python script interface of the scientific data visualization application Mayavi.^[26] These visualizations can be created from one-function call and changed between static and animated via function arguments.

3.2. Performance of the Optimized Controller

We successfully used our PeriSim package to optimize a peristaltic wave controller on the simulated model, shown in Figure 1B. The optimized controller successfully conveyed a ball across the real-world test system in all ten runs. This controller was evolved

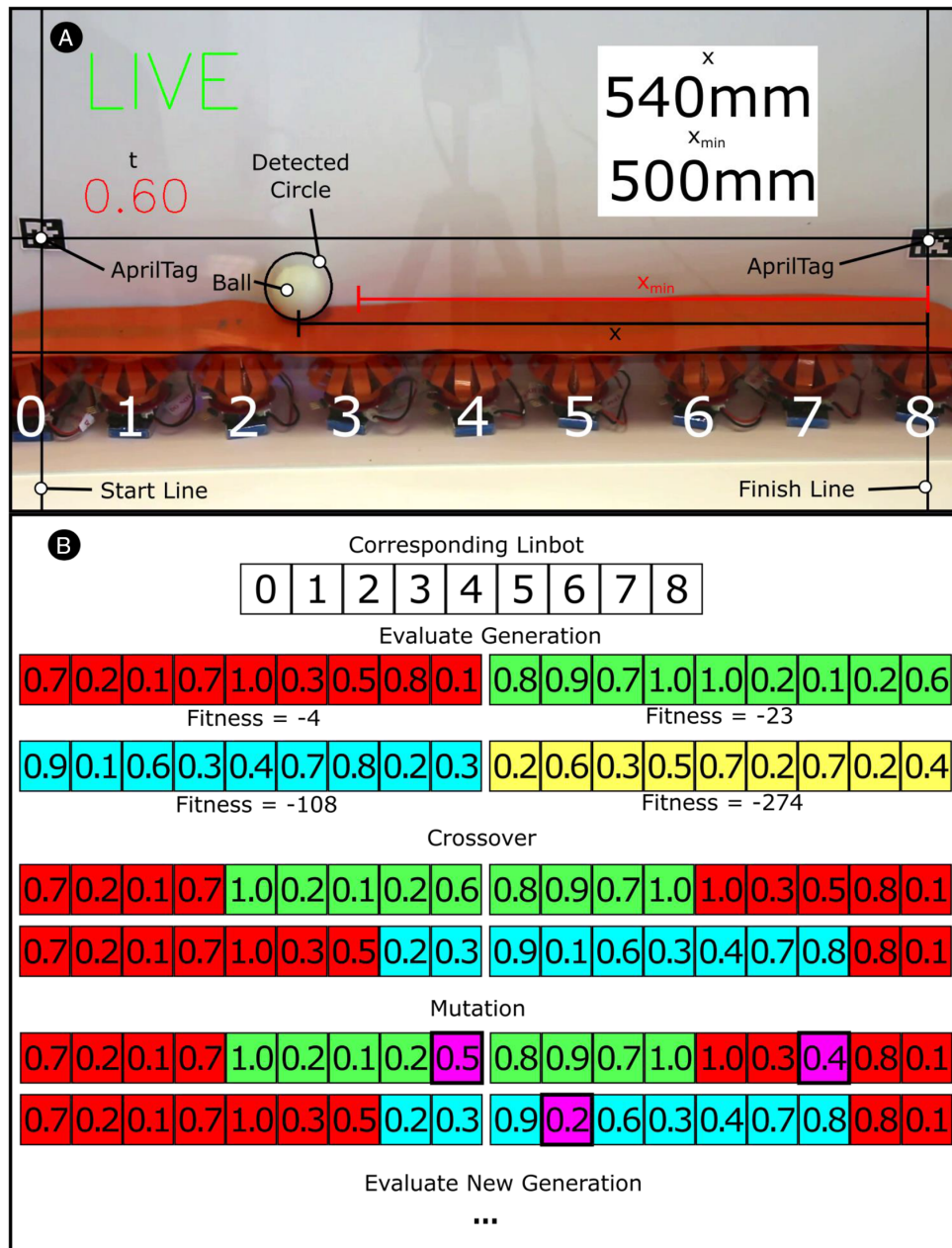


Figure 5. Schematic of genetic algorithm approach. A) A screenshot of data gathering software with annotated information about fitness, f , where $f = -x_{min} - 2t$. The term t is the time taken for the task to finish with a maximum of 10 s, and x_{min} is the minimum distance to the end of the conveyor reached during the test. B) A diagram of a single optimization step of the genetic algorithm. The probability that a genetic code is selected to create part of the next generation is proportional to the fitness of the controller based on the code.

with PeriSim set to a 35% noise level and has an average conveyance time of 1.8 s and so an average fitness of -1.8 . The three controllers with the highest real-world fitness are detailed in Table 1 along with the controller that produced the lowest average conveyance time in successful runs. Video S2, Supporting Information, shows the controller with the highest fitness being used on the peristaltic conveyor. Video S2, Supporting

Information, shows how the control variables are optimized such that the wave pushes the object effectively off the edge.

There were perturbances in the starting position of the ball as well as a chaotic interaction between the ball, the surface, and the sides of the conveyor. These random factors led to every real-world run behaving differently and we did not include them in PeriSim. The performance of the evolved controllers, despite

Table 1. The three selected controllers with the highest fitness (↑) and the controller with the lowest average time of successful runs (↓). The value of interest for each controller is highlighted in bold.

Simulation noise [%]	Simulation fitness	Real-world fitness	Average time for successful runs [s]	Successful run proportion [%]
35	-5.25	-3.61 ↑	1.80	100
35	-5.22	-4.42 ↑	2.21	100
10	-4.74	-28.8 ↑	1.66	90
30	-4.62	-127.2	1.23 ↓	50

these random factors, shows that the controllers are robust to effects that were not explicitly modeled in the simulator.

3.3. Scope for Development

3.3.1. Using PeriSim to Optimize 2D Peristaltic Conveyance

Our simulator can be used for optimizing 2D peristaltic behavior. The degree to which real-world perturbances can disturb the controller will be increased, and it is likely that a different level of noise will be optimal for evolving a controller.

3.3.2. Object Volumes and Shapes

PeriSim can be extended to better model real-world effects. These effects could include the objects no longer being modeled as point masses but instead as masses with a volume that can interact with one another. Improving our simulator in this way also opens up the chance to have objects with different shapes that have more complex interactions with the surface medium. The extension to different shapes will allow for testing that objects can be moved by an existing peristaltic system or aid with the design of new peristaltic systems by modeling the gradients that will be required to roll or slide the objects the system is being designed for. Modeling this extra factor will increase the computational power used by our simulator and so will need to be important to the task being modeled.

3.3.3. Vibrational Sorting

The effect of high-frequency actuation on objects could be added to PeriSim. This would require adding firmness and density characteristics to objects and approximating the effect that these values have when an object is moved down a vibrating slope. Modeling high-frequency actuations will allow for vibrational sorting within PeriSim.

4. Conclusions

Peristaltic tables represent a promising area of research in advanced intelligent systems for the sorting of soft and delicate objects. The control of these tables remains a challenge due to each table being made of a large number of soft, stretchy cells. Machine learning represents an opportunity to create controllers for peristaltic tables, but simulated environments are necessary

for the optimization of these controllers. We presented a novel simulator, PeriSim, that models the fundamental behavior of a peristaltic table using tensor-based mathematics. We used a radical envelope of noise to account for the inherent differences between simulators and the real world when optimizing controllers in PeriSim.

We demonstrated the use of PeriSim for an optimization task using genetic algorithms. This optimization required a large number of simulations to produce optimal results, which was facilitated by PeriSim only modeling fundamental aspects of a peristaltic table. The optimized controller works in the real world despite the simplicity of our simulator.

PeriSim will be of use to anyone exploring the use of peristaltic sorting or conveyance. The radical envelope of noise we used in our simulator will help others produce real-world optimized controllers while avoiding a large reality gap.

Supporting Information

Supporting Information is available from the Wiley Online Library or from the author.

Acknowledgements

This study was supported by the EPSRC via the Robotarium Capital Equipment, and CDT Capital Equipment Grants (EP/L016834/1), and the University of Edinburgh and Heriot-Watt University CDT in Robotics and Autonomous Systems. A.A.S. acknowledges the support from the EPSRC ORCA Hub (EP/R026173/1).

Conflict of Interest

The authors declare no conflict of interest.

Keywords

object conveyance, modular robotics, peristaltic systems, sorting

Received: July 3, 2019

Revised: September 18, 2019

Published online: October 28, 2019

- [1] M. Stommel, W. L. Xu, P. P. K. Lim, B. Kadmury, in *Robot Intelligence Technology and Applications 3: Results from the 3rd International Conference on Robot Intelligence Technology and Applications*, (Eds: J. Kim, W. Yang, J. Jo, P. Sincak, H. Myung, Springer International Publishing, Cham **2015**, pp. 605–615.
- [2] S.-R. Kim, D.-Y. Lee, J.-S. Koh, K.-J. Cho, in *2016 IEEE Int. Conf. Robotics Automation*, IEEE, Piscataway, NJ **2016**.
- [3] A. A. Stanley, A. M. Okamura, *IEEE Trans. Haptics*, **2015**, *8*, 20.
- [4] P. Schoessler, D. Windham, D. Leithinger, S. Follmer, H. Ishii, in *The ACM Symp. User Interface Software Technology*, Charlotte, NC, USA **2015**.
- [5] R. Hashem, B. Smith, D. Browne, W. Xu, M. Stommel, in *IEEE 14th Int. Workshop Advanced Motion Control*, IEEE, Piscataway, NJ **2016**.
- [6] R. M. McKenzie, M. E. Sayed, M. P. Nemitz, B. W. Flynn, A. A. Stokes, *Soft Robotics*, **2019**, *6*, 195.

- [7] C. Uriarte, H. Thamer, M. Freitag, K.-D. Thoben, *Logistics J.: Proc.* **2016**, https://doi.org/10.2195/lj_Proc_uriarte_de_201605_01.
- [8] Intralox, Activated Roller Belt (ARB), **2019**.
- [9] M. Stommel, W. L. Xu, in *IEEE Int. Conf. Mechatronics*, IEEE, Piscataway, NJ **2015**.
- [10] M. Stommel, W. Xu, *IEEE Trans. Autom. Sci. Eng.* **2016**, *13*, 858
- [11] Z. Deng, M. Stommel, W. Xu, *IEEE/ASME Trans. Mechatron.* **2016**, *21*, 1702.
- [12] T. W. Manikas, K. Ashenayi, R. L. Wainwright, *IEEE Instrum. Meas. Mag.* **2007**, *10*, 26.
- [13] E. Yoshida, S. Murata, A. Kamimura, K. Tomita, H. Kurokawa, S. Kokaji, *J. Robot. Mechatron.* **2003**, *15*, 227.
- [14] N. Cheney, R. MacCurdy, J. Clune, H. Lipson, in *Proc. Genetic Evolutionary Computation Conf.*, **2013**.
- [15] N. Jakobi, P. Husbands, I. Harvey, presented at *Proc. Third European Conf. Artificial Life*, Granada, Spain, June, **1995**.
- [16] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, R. Webb, in *IEEE Conf. Computer Vision Pattern Recognition*, IEEE, Piscataway, NJ **2016**.
- [17] P. Abbeel, M. Quigley, A. Y. Ng, *Proc. 23rd Int. Conf. Machine Learning*, ACM, New York, NY, **2006**.
- [18] K. Bousmalis, G. Trigeorgis, N. Silberman, D. Krishnan, D. Erhan, Domain Separation Networks, *Proc. 30th Int. Conf. Neural Information Processing Systems*, Barcelona, Spain **2016**.
- [19] N. Jakobi, *Adapt. Behav.* **1997**, *6*, 325.
- [20] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga and A. Lerer, *31st Conf. Neural Information Processing Systems*, Long Beach, CA, USA **2017**.
- [21] G. Bradski, *Dr. Dobb's J. Softw. Tools* **2000**, *120*, 122.
- [22] K.-F. Man, W. K.-S. Tang, S. Kwong, *IEEE Trans. Industr. Electron.* **1996**, *43*, 519.
- [23] J. Wang, E. Olson, in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots Systems*, IEEE, Piscataway, NJ **2016**.
- [24] J. Illingworth, J. Kittler, *IEEE Trans. Pattern Anal. Mach. Intell.* **1987**, *9*, 690.
- [25] Mayavi, mlab: Python scripting for 3D plotting, Mayavi, [Online]. [https://docs.enthought.com/mayavi/mayavi/mlab.html](https://docs enthought.com/mayavi/mayavi/mlab.html). (accessed: September 2019).
- [26] P. Ramachandran, G. Varoquaux, *Comput. Sci. Eng.* **2011**, *13*, 40.

4.3 Supplemental Videos

For the supplemental videos for this publication scan this QR code:



4.4 Conclusion

This chapter details the design and use of a peristaltic table simulation that I created. This simulation is simple and computationally cheap but is also capable of being used for optimisation thanks to the radical envelope of noise used within it. Table 1 shows the performance of the controller that I optimised within my simulation. The table contains information on the three controllers with the highest real-world fitness, their real-world fitnesses are shown in bold with an upwards arrow to show that a greater value suggests better performance for that metric. I also included information on the controller with the shortest average time for successful runs, the average time per successful run is highlighted in bold with a downwards arrow to show that a smaller value suggests greater performance for that metric. This "fastest" controller shows that a real-world controller could be selected based on a metric other than fitness for example: if speed is desired over reliability.

The simulation contains only the elements that I deemed key to the simulation of a peristaltic table rather than aiming for a perfect model. This approach required simplifications, for example the objects are modelled as always being in contact with the surface. Such simplifications may reduce the performance of PeriSim optimised controllers. The system requires tuning to any new peristaltic system which could be time consuming. I have not created any inbuilt calibration assistance for the simulator.

I have made the simulator available as a python package called PeriSim that can be installed with pip [56]. By sharing this simulator I hope to assist with the further development of peristaltic tables as industrial machinery.

This simulation was useful for my further work designing sorting controllers for my adaptive sorting system. I made a simulated version of my larger peristaltic table, detailed in Chapter 5, which allowed me to test my controllers faster and allowed me to create machine learning based control.

Chapter 5

A Modular Robotic Adaptive Sorting System

5.1 Introduction

In order to demonstrate the potential of a distributed, peristaltic table for sorting I needed to construct and control a modular robotic sorting system using my Linbots and PeriSim. My modular robotic sorting system needed to provide sorting across a range of criteria and be able to handle a range of objects. My sorting system also needed to be able to handle more than one object at once.

5.1.1 Design and Fabrication

I assembled 25 Linbots such that I could form them into a 5x5 grid to build the sorting system. I chose to continue using acetate as a surface to connect the Linbots. I spray painted the acetate surface matt white to provide good contrast against most possible cargo objects. The contrast makes it easier to track the objects using machine vision implementations.

I chose to use tomatoes for cargo objects as they highlight the ability of the system to sort and convey real-world objects, particularly because agricultural sorting is a potential application of my work. The non-quite spheroid shape of tomatoes and the fact that they can be stable in some poses shows that my sorting system doesn't need perfect spheres or sliding objects to work. Tomatoes also have fairly consistent colouring which makes tracking easier. Cherry tomatoes were the type of tomato with the best size and mass for being used on the sorter.

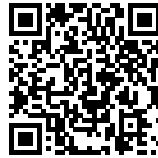
5.1.2 Controlling the Sorter

I created new low and high level control for my adaptive sorting system such that it could perform new task like path-planning and parallel sorting. I continued using a state machine controller to create peristaltic waves but needed to add in more inter-robot communication in order to guarantee that multiple objects moving on the sorter would not interfere with one another. I also added a method for controlling the distance a wave would travel for instances where a cargo object did not need to be moved off of the sorting table. As discussed in Section 2.3.7 the stochastic nature of peristaltic control

and difficulty modelling the surface-object interactions makes standard path planning algorithms difficult to implement. I used the probabilistic automaton control, detailed in Section 2.3.7, which was designed for a centrally controlled system and so I needed to alter it to work on my system.

5.1.3 An Adaptive Sorter Made from Linbots

I built my adaptive sorting system and detailed it in a paper that I have submitted to *Soft Robotics*, where it is in review. Scan the QR code below for a short video summary of the paper.



5.2 Modular Robotic Sorting System Publication

A Modular Robotic Sorting Table

Ross M. McKenzie^{1,2}, Jamie O. Roberts^{1,2}, Mohammed E. Sayed¹, and Adam A. Stokes^{1*}

¹School of Engineering, Institute for Integrated Micro and Nano Systems, The University of Edinburgh, Scottish Microelectronics Centre, Alexander Crum Brown Road, King's Buildings, Edinburgh, UK, EH9 3FF

²EPSRC CDT in Robotics and Autonomous Systems, Edinburgh Centre for Robotics, Edinburgh, UK

* Author to whom correspondence should be addressed.

Keywords: Modular robotics, sorting, recycling, soft robotics

Abstract

Applications such as recycling and warehouse fulfilment require the rapid sorting and grading of a wide variety of objects. Current high-throughput industrial sorting methods are designed for a narrow range of objects to be sorted and sorting criteria to be used. Robotic, peristaltic conveyance can be used for the parallel sorting of a wide variety of objects with different criteria. Peristaltic sorting can also handle delicate and non-rigid objects. Current implementations of peristaltic sorting have been limited by their reliance on central systems to control many actuators. We present a peristaltic sorting table that uses 25 modular robots and a distributed sorting algorithm. This table uses fully distributed algorithms and is capable of changing sorting and conveying behaviour without needing a change to the hardware.

Summary

An adaptive sorting table made from modular robots using distributed control to sort varied objects via multiple criteria.

1 Introduction

1.1 There is a need for the parallel sorting of heterogeneous objects

Current industrial sorting systems allow for low error, high throughput sorting of products [1]. Many of these systems do not process individual objects quickly but instead rely on processing a large number of objects in parallel within the same sorting machine. The downsides of these systems is that they are designed for a narrow range of objects and sorting criteria. Multiple industrial and civil processes require the sorting of objects of various types via multiple criteria.

Profitable recycling requires a good level of purity in output materials which is difficult to obtain from the mixed, disorganised inputs. Fully automated separation of products requires a web of interdependent processing steps, even if the recycling is sorted into bins by the person disposing of it [2]. Pellegrinelli (2018) [3] points out that current implementations rely on hand sorting due to a lack of flexible automated systems capable of adapting to large changes in the composition of the recycling input stream. Pellegrinelli goes on to suggest a theoretical macro-sorting system using universal gripping robots beside a conveyor, this system is flexible but relies on individual item placement by robot arms.

The shipping of groceries or other e-commerce products also requires the sorting of very different objects with little room for error. Even with organised input streams, the difficulty of handling a variety of object shapes means that automation remains a key challenge [4]. The existing research into automating e-commerce fulfilment is focused on pick and place arms due to their flexibility [4] [5]. Agricultural products can require separation from foreign matter after harvest and before any grading [6] requiring multiple sorting stages and machines. A single machine capable of removing foreign material and grading items could provide a more efficient alternative. Port authorities need to randomly test and grade cargo in order to make sure that the correct import codes are being applied, for example in the UK rice must be graded by a variety of criteria [7]. The sample sizes of these tests could be increased given a system flexible enough to sort and grade a variety of objects.

An adaptive, parallel sorting system could provide a mixture between the flexibility of pick and place systems while maintaining the relatively low cost per throughput of mechanical sorting systems.

1.2 Peristaltic motion can be used for adaptive sorting of delicate and soft objects

Peristaltic tables consist of movable cells that are interlocked [8] or connected by a flexible surface [9]. Multiple designs for peristaltic cells have been suggested including self-folding sheets [10], a pneumatic board which can be jammed in a configuration [11], or direct motor linkages [8]. The movement of these cells allows for a controllable heights and gradients, and the production of peristaltic waves. A peristaltic wave is a coordinated motion across multiple cells that moves an object perpendicular to the direction of motion of the cell [9]. The movement of the object is caused by the object accelerating down the gradient produced by the table and as the object moves the table

1
2 changes shape to keep the object on a downward slope, producing a wave-like pattern. Using these
3 peristaltic waves, the position of objects on the peristaltic table can be controlled.
4

5 By adding sensors to the system these objects can be classified and then moved into desired areas for
6 the purpose of sorting. Hashem, et. al. [12] (2016) demonstrated the use of a peristaltic table for
7 moving an object along a desired path. In their work the authors also demonstrate that the table could
8 move larger flat objects suggesting that a peristaltic table has the potential for sorting a variety of
9 objects.
10

11
12 This sorting can be performed on objects that are too slippery or round to be moved reliably by
13 current 2D sorting systems surfaces, that usually rely on rollers pushing against surfaces that are rigid
14 and flat [13] [14]. The flexible surface used in many peristaltic tables can isolate the functional
15 hardware from the cargo being carried by the table, allowing for the sorting of hazardous objects, wet
16 objects [9], liquids [15] or powders [16].
17

18
19 There has been industrial interest in peristaltic tables for sorting and conveyance. The Wavehandler
20 by Festo [17] consists of peristaltic actuators that can be connected to form a peristaltic table. The
21 ability to easily connect and disconnect the actuators allows for easier scaling of the sorting system
22 although their need to be interlocked does mean the size of the table is strictly defined for the number
23 of actuators. The actuators of the wavehandler can also perform distributed algorithms in order to
24 route pneumatic power to the desired cells based on commands from a central system connected to a
25 camera. This work shows the industrial interest in systems that can combine functionality as they
26 push the benefits of “conveying and sorting in one”.
27

28
29 As noted by Festo in their work on the Wavehandler, in future systems it would be beneficial for the
30 sub-systems to be able to perform some tasks autonomously and be capable of decision making to
31 avoid reliance on central control [17]. This decoupling from central control would increase the
32 scalability of an adaptive sorting system while also splitting the computation required for parallel
33 sorting across the involved robots. To create decentralised intelligence in a peristaltic sorting system
34 we can use our modular robots, the Linbots [18], as peristaltic cells with their own sensing,
35 communication and decision making.
36

37 38 **1.3 Modular robots allow for scalability and distributed control**

39
40 Modular robots are robots that can be linked together in order to form collectives, they usually rely
41 on individual sensing, communication and decision making but need to cooperate with others such
42 that the collective can perform useful work [19]. Modular robots share many advantages with swarm
43 robots as both utilise multiagent behaviour [20], with modular robots emphasising physical linking
44 between units. Using modular robots as peristaltic cells comes with the benefit of inherent scalability
45 as there is no central system limiting the number of sensors or actuators that can be in use [21].
46 Multiple modular units are also more robust than a single, larger system [22] [23] as a component
47 failure in one robot can be compensated by other robots within the system and is less likely to cause a
48 systemwide failure. Modular robot collectives can also be reconfigured for different tasks [24] [25]
49 this reconfigurability can either be through self-reconfiguration [22] or human intervention to rebuild
50 the collective.
51

52
53
54 Liu, et. al. (2008) [26] demonstrate that modular robots can be used to create a scalable system that
55 relies on distributed sensing and control to function. This work highlights the benefit of a modular
56 robotic approach for a system that requires parallel behaviour.
57
58
59

1.4 System control

An adaptive sorter based on a peristaltic table will be difficult to reliably control due to the reliance on multiple agents working together and the indirect method of moving cargo [27]. The control will only become more difficult if the objects being moved do not roll or slide in a manner close to ideal which will be the case for many potential types of cargo.

In our previous work [18] we used a finite state machine controller to produce peristaltic wave patterns. These functioned well for single direction movements and were easy to implement on microcontrollers. Our implementation lacked any higher-level path planning and did not take into account any obstacles or other objects moving on the peristaltic system.

M. Stommel and W. Xu (2016) [28] suggest a control method based on learning the dynamics of the system and encoding them in a finite state machine. First the system needs to be broken into a set of discrete states (\mathcal{S}) that describe the possible configurations of the system, which can include object poses and actuation states. The system also need a predefined set of actions (\mathcal{A}) which may include peristaltic waves with different parameters. The system is then tested to find the probabilities of the states ($s_r \in \mathcal{S}$) resulting from an action ($a \in \mathcal{A}$) in a state ($s \in \mathcal{S}$). Once the probabilities of resultant states are found for every state action pair an optimisation can be run to find the expected cost (c) of reaching a goal state. They call this optimisation efficient choice and it takes the form of an iterative equation:

$$c(s, a, t + 1) = T + \sum_{s_r \in \mathcal{S}} p(s_r | s, a) \min_{b \in \mathcal{A}} c(s_r, b, t) \quad (1)$$

Where (T) is the cost of the state transition and (t) is a variable tracking the iteration of the equation. The goal state starts with $c = 0$ for all actions. A cost of human intervention (T_f) is also required to cover the systems transitioning to a failure state. This method is similar to value iteration in reinforcement learning [29]. The system then chooses its lowest cost action in each state in order to most efficiently transition to the goal state. This approach does allow for complex control and will function on a range of different hardware implementations making it a flexible control choice for an adaptive sorter. It does however rely on central control and doesn't define any low-level behaviour. We can combine this control with lower level state-machine control from our previous work to get a full control system, we also need to adapt the system to be suitable for distributed control across multiple robots.

In our work we combine the hardware design of peristaltic sorting systems with modular robots to build a proof of concept adaptive sorting system. We then use our existing state machine control combined with probabilistic automaton control to overcome the difficulties of complex movement with a peristaltic system. Our system is capable of sorting by multiple criteria and can move differently shaped objects. Video S1 offers a short summary of the system.

2 System design

2.1 Linbots

To build our adaptive sorting system we used our modular robots called Linbots. Linbots use a voice coil system made from permanent magnets and a wire coil joined together with flexible material that acts as a spring. We can use the voice coil system to cause the Linbots to contract or extend. This shape change allows the Linbots to function as peristaltic cells. The Linbots Are also capable of

1
2 sensing the displacement of their top half relative to their bottom half meaning that they function as
3 tactile sensors and can measure the weight of objects stacked on top of them. The Linbots
4 communicate with one another using a custom circuit using the same coil that they use for actuation
5 as an antenna. As the Linbots are built by hand they exhibit inconsistencies including size and
6 actuation force. We have already demonstrated the Linbots as part of a reconfigurable peristaltic
7 system that functioned as a 9 robot long peristaltic conveyor and could be changed into a 3x3 array
8 for peristaltic sorting. For further information on our Linbot system see Ross McKenzie, et. al.
9 (2019) [18].
10
11

12 **2.2 Design of the adaptive sorter**

13
14 Our adaptive sorter consists of a 5x5 square grid of Linbots with an 80mm grid spacing. Each Linbot
15 is connected to a switch board to allow for easy power resets. We affix the Linbots to both a flexible
16 surface on top of the sorter and to a substrate such that an individual contracting Linbot will pull
17 down the surface. We used Velcro to attach the surface to allow for easy removal and replacing
18 when working on the Linbots beneath. The substrate is held up on spacers to allow for the routing of
19 power wires below it and has gaps so that the wires can pass through it. The full experimental setup
20 is shown in Figure 1. The substrate has two sets of Linbot outlines etched into it to guide Linbot
21 placement. One set have a spacing of 80mm the other have a spacing of 90mm, for this paper we
22 used the 80mm spacing outlines as guides.
23
24

25
26 We made three flexible surfaces for the table out of acrylic and spray painted two of them, one white,
27 the other orange. We weighted the edges of the surface to ensure an outwards gradient at the edges of
28 the table. The substrate and spacers are made from acrylic.
29

30 We manually programmed each Linbot with an address based on the addressing system we
31 developed in our previous Linbot paper [18]. These addresses allow neighbouring Linbots to know
32 the origin and relative direction of commands so that they can respond appropriately to each
33 communication. The addressing system is based on the grid distance of each robot from the top left
34 and bottom left corners of the table and is shown by Figure 2. We programmed the Linbots to
35 calculate whether they are at the edge of the table using their coordinates. As edge Linbots cannot use
36 two neighbours to create a gradient normal to the edge and the cargo will fall off the table when
37 moved parallel to the edge along the edge we do not use any edge Linbots for starting peristaltic
38 waves.
39
40

41 **2.3 Design in simulation**

42
43 We required a simulated model for designing controllers for the sorting table. Using the physical
44 hardware to test each iteration of the software would lead to delays caused by the time taken to reset
45 the system with each new iteration. Instead we used our peristaltic table simulator PeriSim [30].
46 PeriSim can be calibrated to match any peristaltic table system so initially we made sure the
47 behaviour of the simulator matched the behaviour of the real system as closely as possible. This
48 simulator allowed us to quickly iterate on the controller designs as well as run large batches of tests
49 with time acceleration. Our peristaltic table behaves stochastically due to minor differences in cargo
50 shape and placement and surface gradient leading to potentially large differences in behaviour.
51 PeriSim accounts for this using a radical envelope of noise [31], which describes a method of
52 randomly perturbing the underlying parameters of the simulation at each run, in addition the
53 acceleration is perturbed randomly at each timestep with larger accelerations being perturbed more.
54 Through multiple runs with each controller we can see if the controller is robust in a variety of
55 situations allowing for it to be robust to real world perturbations.
56
57
58
59
60

1
2 PeriSim does not model vibrational effects and so we added any vibrational aspects to the controllers
3 after they had been optimised in simulation. We designed each controller in simulation for testing.
4 An example of one of these test simulations can be seen in Video S2.
5

6 7 **2.4 Design of a peristaltic wave controller**

8
9 Peristaltic waves are initiated in response to a weight being placed upon the Linbot. Depending on
10 the experiment the weight can also affect the direction of the wave. When a weight is placed on a
11 Linbot the robots required for a successful movement are queried and locked to this movement if not
12 already locked. This query takes the form of a series of messages and acknowledgements along the
13 entire path that the wave will take. The messages contain information about the direction of the wave
14 and the position of the broadcasting Linbot such that only the desired Linbots will process the
15 message. If an already locked robot is encountered the movement will be paused until the required
16 robot becomes free. When the origin robot receives an acknowledgement that all the required robots
17 are locked it sends a “Go” signal then either extends or vibrates depending on a preset option, see
18 section 2.3.
19

20
21 On detecting a “Go” signal the locked Linbots are programmed to propagate the signal after a delay
22 and then contract. After a set period of contraction, the Linbots return to rest for a time and then
23 extend. This sequence creates a traveling wave in the surface of the sorting table that conveys the
24 object. The delays are set such that the trough of the wave is two Linbots wide. After taking part in
25 the movement the Linbots are briefly dormant and then unlock and return to their rest state.
26 Schematics of the mechanism are shown in Figure 2A-B.
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41

42 **2.5 Destabilising cargo**

43
44 Many agricultural products are not perfectly spheroid and have some flat surfaces leading to poses in
45 which they are stable and will not roll. The tomatoes we use as test cargo exhibit this behaviour,
46 detailed in Figure 3A. This means that standard peristaltic waves will not be able to start moving any
47 cargo that is at rest in a stable pose. In order to move cargo to an unstable pose we programmed the
48 sorter to shake the cargo with a low-frequency vibration. By lowering the frequency of the vibration,
49 the Linbot spends more time actuating between changes and produces higher amplitude oscillations.
50 This high amplitude vibration causes the cargo to bounce and turn leading to it eventually landing in
51 an unstable pose.
52
53

54
55 The frequency of the vibrations needed to be carefully tuned as having too high an amplitude could
56 cause the cargo to get thrown out of the peristaltic wave while too low an amplitude would not move
57 the cargo out of a stable pose. The shape and mass of the cargo affects the range of suitable
58
59
60

1
2 amplitudes for destabilisation. The Linbots also produced slightly different amplitude vibrations from
3 the same command inputs due to shape inconsistencies. To increase the reliability of these
4 destabilisations we programmed the Linbots to use two frequencies of vibration. Each destabilisation
5 uses first 54Hz and then decreases to 31Hz. Any cargo that can be destabilised at the higher
6 frequency moves away down the gradient at that point thereby avoiding the higher amplitude
7 oscillations that follow. These vibrations allow for the successful rolling of a variety of shapes and
8 masses of cargo. This system is not perfect as there are some stochastic elements inherent to the rapid
9 bouncing and turning of a non-spherical object so the cargo always has a small chance of being
10 knocked away from the wave or not being destabilised at all.
11

12
13 We programmed the Linbots to use this destabilisation method at the start of each peristaltic wave so
14 that any cargo is then unstable and can then be rolled via the rest of the wave. We use this behaviour
15 for all of our peristaltic waves with tomato cargo.
16

17
18 There are some situations where a sorting table may be unable to use all of its area for transporting
19 objects. To allow for this situation we programmed our sorter with path planning behaviour
20 optimised in simulation.
21

22 **2.6 Path planning control**

23
24 To implement the control method of Stommel and Xu (2016) [28] we defined a number of discrete
25 states that describe the XY position of the cargo on the sorting table. Each state corresponds to a
26 80mm square on the table with the system being in that state if the cargo is within the square. This
27 creates 25 states overall. We then design a number of actions that can be taken in each state these
28 actions are state-machine based peristaltic waves, see Section 2.4, with four directions (up, down,
29 left, and right). We alter the controller to also use two distances (one and two) giving 8 possible
30 actions.
31

32
33 The distance of the wave describes the number of Linbots that will be used in the wave, not including
34 the Linbot starting the wave. We used only two lengths as a wave distance of three or more would
35 carry the object from any point on the table to an edge state or off of the table.
36

37
38 The states that lie on the edge of the table and can only create functional peristaltic waves in the
39 direction away from the interior of the table due to the lack of a Linbot on the opposite side, see
40 Section 2.2. Edge states that are not in the corners only have one action which corresponds to a wave
41 away from the interior of the table with a distance of 1. Corner states have two actions as they have
42 two directions away from the interior of the table.
43

44
45 To find the optimal path with this mapping we used the efficient choice method described in
46 Equation 1. We altered the method such that (T) can have different values for different actions, this
47 was to encourage the system to use shorter peristaltic movements where suitable but to prefer longer
48 distance peristaltic waves over a number of shorter waves. We used $T = 1$ for 1 distance peristaltic
49 waves and $T = 1.5$ for two distance peristaltic waves. We needed to choose a suitable value for (T_f),
50 the higher the value the less risks the system will take for the sake of speed. We chose $T_f = 10$ as it
51 provided a good level of risk avoidance.
52

53
54 To use this controller in a distributed manner both in simulator and the real-world system we took the
55 best actions in each state from the controller and saved them to the memory of the respective Linbot.
56 When the Linbot detects a weight, it communicates with neighbouring robots to execute the best
57 action saved to it.
58

3 Experimental design

3.1 Sorting by mass

When a Linbot detects a weight placed on it, it increments a counter related to the class of the weight it detects. When the counter of a class reaches a threshold, the Linbot sends a message to its neighbours detailing the direction of a desired peristaltic wave, this build-up and discharge behaviour is inspired by action potentials in heart muscle cells [32]. The created wave is specified such that it should move the object off the sorting area in the correct direction. The Linbots all have the same controller and can function in any position within the system once their coordinates are updated.

As the Linbots are handmade there are inconsistencies in their construction that we detailed in our previous Linbot paper [18]. In order to better standardise the mass readings across the robots we calibrate the Linbots before each run with a 7.99g weight.

We used 50 cherry tomatoes separated equally into two mass classes, one for tomatoes with less than 8 grams of mass and one for tomatoes with 8 or more grams of mass. We programmed the sorter to move the $<8g$ tomatoes off of the left side of the sorter and $\geq 8g$ tomatoes off of the bottom side of the sorter. We placed the tomatoes in the centre of the sorter to allow for a fair comparison of errors.

To compare the source of errors we defined the sorting task as the combination of two component tasks: a classification task and a movement task. We considered classification a success if the tomato being sorted was correctly classified as $<8g$ or $\geq 8g$. We considered the movement task a success if the tomato was moved off of the correct side of the table for the classification given to it by the sorter. We considered the combined task successful if both of the previous tasks were successful.

3.2 Parallel Sorting

Due the distributed nature of the sorter, it can convey more than one object at once, shown by Video S3. We do however need to test that our sorting system is capable of handling more than one object at a time while avoiding object collision. We tested parallel sorting by placing the system in a state where two correct sorting movements intersect. We used the weight sorting Linbot controller with the $\geq 8g$ class direction changed to downwards and placed a $\geq 8g$ tomato on the top left (1,3) of the sorter, we simultaneously placed a $<8g$ tomato on the bottom right (3,1) of the sorter. This placement led to the sorter needing to move both tomatoes across the same Linbot in the bottom left (1,1). We recorded how the sorting system handled this situation.

3.3 Sorting by shape

Our shape sorting algorithm exploits the difference in stability between tomato shapes. More spheroid tomatoes are easier to move out of their stable pose. We first classified two tomatoes as as more spheroid and less spheroid by eye. A To sort between different shapes of tomatoes we programmed the Linbots to initially classify the tomato as more spheroid and use a normal peristaltic wave with a 54Hz destabilising vibration to move an object off of the left side of the sorter. If the Linbot detects that there is still a weight upon it after the motion has been completed it now classifies the object as less spheroid and uses 31Hz vibrations to destabilise the tomato before using a peristaltic wave to move it off of the right side of the table.

3.4 Vibrational conveyance

1
2 Peristaltic waves cannot move objects that do not slide or roll down gradients. Objects that would
3 otherwise not slide or roll across a surface can be moved via biased vibrations pushing them in the
4 direction of travel [33]. For our adaptive sorting system this vibrational “pushes” are biased by
5 changing the gradient of the surface via contracting and extending Linbots. Linbots can change their
6 height while vibrating using PWM. As the object moves across the surface the Linbots need to
7 cooperate in the same manner as a peristaltic wave in order to keep a consistent gradient direction.
8 Therefore, the vibrational motion used for our sorter is a hybrid of vibrational and peristaltic motion
9 and is shown in detail by Figure 3B. As the behaviour is similar to a peristaltic wave it can be
10 completed using the same controller behaviour as mentioned in section 2.4 but with the Linbots
11 vibrating at 22Hz between contraction and extension steps.
12
13

14 3.5 Vibrational separation

15
16 Objects of different shape and size move differently under the effects of vibrational conveyance
17 which can be used for separating those objects [34]. We tested the sorter separating a granular
18 medium, salt, from a piece of cargo, a tomato. We programmed the Linbot beneath the weight to
19 vibrate at 144Hz while one neighbouring Linbot contracts. At 144Hz the amplitude of the vibrations
20 is small enough to produce no movement in the tomato while still causing the salt to move down the
21 gradient.
22
23

24 3.6 Path planning

25
26 To add complexity to the path planning task we added a barrier to stop the ball from passing across
27 the two central-line Linbots towards the top of the table, (2,3) and (2,4), see Figure 4A for the
28 physical setup. The Linbots under the barrier are off and so do not communicate with their
29 neighbours. The coordinates of the robots are kept the same. We then set the start and goal states on
30 either side of the barrier, states (1,3) and (3,3) respectively, such that multiple actions and changes in
31 direction are required to move the cargo to the goal state.
32
33

34 We modelled this new setup in simulation and used it to map the probability distributions of resultant
35 states from all actions taken in the non-edge, non-barrier states. Using our altered efficient choice
36 method, we calculated the expected cost to goal and best actions for each state. To run a real-world
37 test of the controller we programmed the Linbots with the best action for their respective state. We
38 used a tomato as the cargo item. We placed the tomato on the start state and tracked its movement.
39
40

41 4 Results and Discussion

42 4.1 Sorting by weight

43
44 The combined task was successful for 41/50 (82%) of the tests. The classification task was successful
45 for 45/50 (90%) of the tests. The movement task was successful for 45/50 (90%) of the tests. One test
46 failed both the classification and movement tasks. These results and the classification matrix are
47 shown in Figure 5A.
48
49

50
51 We plotted the distribution of the errors against the mass of the object, relative to the class threshold,
52 in Figure 5B. The errors from the classification task are mainly clustered around the threshold which
53 is to be expected as small changes in placement of the object can affect the output of the hall effect
54 sensor, leading to slightly incorrect mass measurements. There is one classification error far from the
55 threshold which was likely caused by a mechanical failure. There are occasions when one of the two
56 plastic sheaths used to hold the permanent magnets and the coil can catch on the seam of another
57
58
59
60

1
2 leading to extra resistance to movement under a weight. This shows that the reliability of the sorter
3 could be improved via a mechanical redesign to avoid such catches.
4

5 The errors from the movement task are more relatively evenly spread. These failures were expected,
6 see Section 2.5. The bias towards lower mass tomatoes could suggest they are more vulnerable to
7 these failures.
8

9 10 **4.2 Parallel Sorting**

11 We observed that the objects were correctly classified and sorted off-of the correct edges. The order
12 in which the objects were sorted depended on which movement could lock the Linbot at their
13 interception point first. The <8g tomato used in the experiment moved off of the left-hand side of the
14 sorter. After the Linbot at the interception point of the two movements emerged from dormancy the
15 >=8g tomato was moved off of the bottom of the sorter. The experiment can be seen in Video S4.
16 Snapshots from Video S4 can be seen in Figure 6A.
17
18

19 20 **4.3 Sorting by shape**

21
22 The results are shown in Video S5. A series of snapshots of Video S5 are shown by Figure 6B. The
23 video shows that the lower amplitude vibrations do not destabilise the less spheroid tomato but do
24 destabilise the more spheroid tomato. Both tomatoes are move successfully off of the correct sides of
25 the sorter. The less spheroid tomato can be seen moving during the first destabilisation attempt, this
26 is due to it being vibrationally conveyed, see Section 3.4.
27
28

29 30 **4.4 Vibrational conveyance**

31 The experiment can be seen in supplemental Video S6. A series of snapshots of Video S6 are shown
32 by Figure 3C. The disk that does not roll or slide down the gradient normally is moved via
33 vibrational conveyance. The disk covered 100mm in 8s giving it an average speed of 12.5mms^{-1} . The
34 disk can only be controlled with a resolution of one peristaltic cell width (80mm) as no gradient
35 changes can be made at a smaller resolution than one peristaltic cell. This can be seen by the disk
36 moving laterally with relation to the gradient direction. The disk moves to the left side of the channel
37 travelled by the vibrational, peristaltic wave. This is caused by the amplitude of the vibrations being
38 larger directly above the Linbots which pushes the disk away from the centre of the channel.
39
40

41 42 **4.5 Vibrational separation**

43 The experiment can be seen in supplemental Video S7. Snapshots of the experiments are seen in
44 Figure 7. The vast majority of the salt was successfully separated from the tomato with only a few
45 grains remaining stuck to the surface near the tomato. Most of the salt movement takes place within
46 seconds of the vibration starting. The weight of the tomato creates an area where the amplitude of the
47 vibrations is reduced. The salt is only slowly removed from this area, slowing the separation process.
48 The grains that did not move down the surface have adhered to the surface, likely due to some
49 moisture from the air or tomato.
50
51

52 53 **4.6 Path planning**

54 We optimised the planned path in simulation to find the resultant state probability distributions for
55 each state-action pair, an example of these probability distributions is shown in Figure 4B. We found
56 the optimal path for the cargo using our altered efficient choice method, the expected cost to goal and
57
58
59
60

1
2 best actions for each state are shown by Figure 4C. The controller suggested an optimal path which
3 matched the path we expected. The result of the real-world test can be seen in Video S8. The path
4 taken by the tomato can be seen in Figure 4D. The path planning controller successfully moved the
5 tomato from the start state to the goal state around the block along the most efficient path. The main
6 time cost it the time taken for the Linbots to come out of dormancy and check if the tomato is in their
7 respective state. We expected the tomato to remain within half a peristaltic cell width of the planned
8 path, this was not the case and at one point the tomato moved beyond that distance due to a large kick
9 during destabilisation, the simulation suggested. The tomato did roll back to the most likely resultant
10 state afterwards therefore the actions taken by the sorter were still optimal.
11
12

13 **4.7 Scope for development**

14 **4.7.1 Increased Linbot consistency**

15 The current method for building Linbots is advantageous for prototyping a system as they can be
16 built mostly by hand. In order to increase the consistency of the Linbots in shape and behaviour other
17 methods such as machine folding could be used. Being more consistent would remove the need for
18 calibration and make the system behave closer to the simulated model.
19
20
21
22

23 **4.7.2 Larger gradients**

24 Producing larger gradients on our sorting system would allow for faster cargo conveying and easier
25 cargo destabilisation. By making the Linbots more tightly packed we could increase the gradients on
26 the surface of the sorter with the trade off of making the sorting area smaller.
27
28
29
30

31 **4.7.3 Non-rectangular collectives**

32 The current collective awareness algorithm only allows for rectangular collectives and a rectangular
33 table laid out in a squared grid pattern. Other patterns such as hexagonal or diamond could allow for
34 different wave patterns and ranges of motion. The collective awareness algorithm could be adapted to
35 work with these other collective shapes.
36
37

38 **4.7.4 Chladni patterns**

39 Chladni patterns are lines of stationary nodes formed by sound waves passing through a flat plane,
40 these patterns can be used to control the position of objects [35]. If a Linbot is rigidly connected to a
41 level surface it should theoretically be able to produce Chladni patterns. If a sorting system could
42 produce these patterns it should allow for the movement of multiple objects at a resolution
43 significantly smaller than the size of each peristaltic cell.
44
45

46 **4.7.5 Liquid transport**

47 The surface of our sorter is waterproof and could be used to convey liquids with peristaltic waves.
48 The experimental setup, however, would need to be altered to achieve a reasonable level of safety.
49 Extra precautions would need to be taken to guarantee no liquid escaping the system in the case of
50 mistake or malfunction.
51
52

53 **4.7.6 Movement locking avoidance**

54 Currently the system can enter a state where every movement is blocked by another and the system
55 becomes locked and unable to move. This state is entered when two movements moving in opposite
56
57
58
59
60

1
2 directions collide or when a minimum of 4 movements in a rectangular pattern start at similar times
3 and each block each other. Some of these cases can be solved by allowing one movement to take
4 place before the others, or by moving the cargo laterally. However, a more general solution to
5 locking will require online, distributed path planning.
6

7 8 **4.7.7 Movement timings**

9 Currently when movements intersect the second movement must wait for the first movement to
10 finish before starting. It is, however, possible to estimate the time the first movement will take to pass
11 and start the second movement earlier while still avoiding a collision. To achieve this overlapping
12 movement, reserved Linbots will need to record how far along a movement they are and have an
13 internal estimate of when the movement will pass through them. When another movement attempts
14 to lock the Linbot that Linbot could become reserved to both movements at different times and can
15 then pass the estimated time of the first movement back to the origin Linbot of the second movement.
16 This origin Linbot can then delay starting the second movement to avoid collision.
17
18

19 20 **5 Conclusions**

21
22 In this paper we present a proof of concept adaptive sorting system. This system is a peristaltic table
23 with 25 Linbots acting as peristaltic cells. We have demonstrated that the table is able to convey
24 multiple types of objects and sort for multiple criteria without changing the hardware. The sorting
25 system is also capable of parallel sorting. We have also demonstrated higher level control with a
26 pathfinding algorithm that is robust to the stochastic nature of peristaltic motion. We developed much
27 of the control in simulation which then functioned when controlling the real-world system.
28
29

30 We note that the Linbots are prototype systems and lack the consistency of mass-produced
31 components. This leads to the sorter having a non-flat surface and having slightly different actuation
32 and sensing behaviour in different areas of the sorter. These inconsistencies make the sorter behave
33 differently to the simulated model and introduce potential points of failure. The controllers optimised
34 in simulation did, however, function well on our real-world system due to the simulators in-built
35 radical envelope of noise.
36

37
38 Our work represent a possible future path for a new type of sorting system capable of replacing
39 multiple mechanical sorters. These new adaptive sorting systems can also handle a wider variety of
40 objects than mechanical sorters and could be used in situations that currently rely on the picking of
41 objects by hand.
42

43 44 **References**

- 45
46
47
48
49 [1] P. Chen and Z. Sun, "A review of non-destructive methods for quality evaluation and sorting of agricultural
50 products," *Journal of Agricultural Engineering Research*, vol. 49, no. Supplement C, pp. 85-98, 1991.
51 [2] S. P. Gundupalli, S. Hait and A. Thakur, "A review on automated sorting of source-separated municipal solid waste
52 for recycling," *Waste Management*, vol. 60, pp. 56-74, 2017.
53 [3] S. Pellegrinelli, "Configuration and reconfiguration of robotic systems for waste macro sorting," *The International
54 Journal of Advanced Manufacturing Technology*, vol. 102, no. 9, pp. 3677-3687, June 2019.
55 [4] C. Liang, K. Chee, Y. Zou, H. Zhu, A. Causo, S. Vidas, T. Teng, I. Chen, K. Low and C. Cheah, "Automated Robot
56 Picking System for E-Commerce Fulfillment Warehouse Application," 2015.
57
58
59
60

- 1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
- [5] A. Zeng, S. Song, K. Yu, E. Donlon, F. R. Hogan, M. Bauza, D. Ma, O. Taylor, M. Liu, E. Romo, N. Fazeli, F. Alet, N. C. Dafle, R. Holladay, I. Morena, P. Qu Nair, D. Green, I. Taylor, W. Liu, T. Funkhouser and A. Rodriguez, "Robotic Pick-and-Place of Novel Objects in Clutter with Multi-Affordance Grasping and Cross-Domain Image Matching," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018.
- [6] R. Pellenc and J. Gialis, "Sorting table with sorter rolls for elimination of foreign matter remaining mixed in a harvest of small fruit". 2007.
- [7] HM Revenue & Customs, *Classifying rice for import and export - GOV.UK*, 2015.
- [8] S. Follmer, D. Leithinger, A. Olwal, A. Hogge and H. Ishii, "inFORM: dynamic physical affordances and constraints through shape and object actuation.," in *UIST*, 2013.
- [9] M. Stommel, W. L. Xu, P. P. K. Lim and B. Kadmiry, "Soft Peristaltic Actuation for the Harvesting of Ovine Offal," in *Robot Intelligence Technology and Applications 3: Results from the 3rd International Conference on Robot Intelligence Technology and Applications*, J. Kim, W. Yang, J. Jo, P. Sincak and H. Myung, Eds., Cham, Springer International Publishing, 2015, pp. 605-615.
- [10] S.-R. Kim, D.-Y. Lee, J.-S. Koh and K.-J. Cho, "Fast, compact, and lightweight shape-shifting system composed of distributed self-folding origami modules.," in *ICRA*, 2016.
- [11] A. A. Stanley and A. M. Okamura, "Controllable Surface Haptics via Particle Jamming and Pneumatics.," *IEEE Trans. Haptics*, vol. 8, no. 1, pp. 20-30, 2015.
- [12] R. Hashem, B. Smith, D. Browne, W. Xu and M. Stommel, "Control of a soft-bodied XY peristaltic table for delicate sorting.," in *AMC*, 2016.
- [13] C. Uriarte, H. Thamer, M. Freitag and K.-D. Thoben, "Flexible automation of logistics processes by means of modular robotic and material flow systems," 2016.
- [14] Intralox, *Activated Roller Belt (ARB)*, 2019.
- [15] T. Tone and K. Suzuki, "An Automated Liquid Manipulation by Using a Ferrofluid-Based Robotic Sheet.," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 2814-2821, 2018.
- [16] Y. Yamada, K. Ashigaki, S. Yoshihama, K. Negishi, K. Kato and T. Nakamura, "Triangular cross-section peristaltic conveyor for transporting powders at high speed in printers," *Advanced Robotics*, vol. 32, no. 12, pp. 646-658, 2018.
- [17] Festo, *WaveHandling*, 2013.
- [18] R. M. McKenzie, M. E. Sayed, M. P. Nemitz, B. W. Flynn and A. A. Stokes, "Linbots: Soft Modular Robots Utilizing Voice Coils," *Soft Robotics*, vol. 6, no. 2, pp. 195-205, April 2019.
- [19] P. Stone and M. Veloso, "Multiagent Systems: A Survey from a Machine Learning Perspective," *Autonomous Robots*, vol. 8, no. 3, pp. 345-383, June 2000.
- [20] Z. Shi, J. Tu, Q. Zhang, L. Liu and J. Wei, "A Survey of Swarm Robotics System," in *Advances in Swarm Intelligence*, Berlin, Heidelberg, 2012.
- [21] S. Murata, K. Kakomura and H. Kurokawa, "Toward a scalable modular robotic system," *IEEE Robotics Automation Magazine*, vol. 14, no. 4, pp. 56-63, December 2007.
- [22] M. Yim, W.-m. Shen, B. Salemi, D. Rus, M. Moll, H. Lipson, E. Klavins and G. S. Chirikjian, "Modular Self-Reconfigurable Robot Systems [Grand Challenges of Robotics]," *IEEE Robotics & Automation Magazine*, vol. 14, no. 1, pp. 43-52, March 2007.
- [23] J. Clune, J.-B. Mouret and H. Lipson, "The evolutionary origins of modularity," *Proceedings of the Royal Society of London B: Biological Sciences*, vol. 280, no. 1755, 2013.
- [24] C. H. Belke and J. Paik, "Mori: A Modular Origami Robot," *IEEE/ASME Transactions on Mechatronics*, vol. 22, no. 5, pp. 2153-2164, October 2017.
- [25] J. Neubert, A. Rost and H. Lipson, "Self-Soldering Connectors for Modular Robots.," *IEEE Trans. Robotics*, vol. 30, no. 6, pp. 1344-1357, 2014.
- [26] G. Liu, S. Abdul and A. A. Goldenberg, "Distributed control of modular and reconfigurable robot with torque sensing.," *Robotica*, vol. 26, no. 1, pp. 75-84, 2008.
- [27] M. Stommel and W. L. Xu, "Qualitative control of soft robotic peristaltic sorting tables," in *2015 IEEE International Conference on Mechatronics (ICM)*, 2015.
- [28] M. Stommel and W. Xu, "Optimal, Efficient Sequential Control of a Soft-Bodied, Peristaltic Sorting Table," *IEEE Transactions on Automation Science and Engineering*, vol. 13, no. 2, pp. 858-867, April 2016.
- [29] T. Bountourelis, "Total Expected Discounted Reward MDPs: Value Iteration Algorithm," vol. Wiley Encyclopedia of Operations Research and Management Science, 2011.
- [30] R. M. McKenzie, J. O. Roberts, M. E. Sayed and A. A. Stokes, "PeriSim: A Simulator for Optimising Peristaltic Table Control," *Advanced Intelligent Systems*, In Press.

- 1
2 [31] N. Jakobi, "Evolutionary Robotics and the Radical Envelope-of-Noise Hypothesis," *Adaptive Behavior*, vol. 6, no.
3 2, pp. 325-368, 1997.
4 [32] N. Cheney, J. Clune and H. Lipson, "Evolved Electrophysiological Soft Robots," *ALIFE*, 2014.
5 [33] A. I. Ribic and Ž. V. Despotovic, "High-Performance Feedback Control of Electromagnetic Vibratory Feeder,"
6 *IEEE Transactions on Industrial Electronics*, vol. 57, no. 9, pp. 3087-3094, Sep. 2010.
7 [34] R. Cohen-Alloro and R. Cuvillier, "Vibrating table for the gravimetric separation of fine particles". 1975.
8 [35] Q. Zhou, V. Sariola, K. Latifi and V. Liimatainen, "Controlling the motion of multiple objects on a Chladni plate,"
9 *Nature Communications*, vol. 7, no. 1, pp. 12764--, 2016.
10
11
12
13
14
15

16 Acknowledgments

17 Author contributions

18
19
20 RM: Building the sorter and designing the controllers used. AS: lead advisor and primary editor of
21 the manuscript. JR: Assistance with the experiments and data analysis. MS: Assistance with the
22 experiments.
23
24
25

26 Funding

27
28
29 This study was supported by EPSRC via the Robotarium Capital Equipment, and CDT Capital
30 Equipment Grants (EP/L016834/1), and the University of Edinburgh and Heriot Watt University CDT
31 in Robotics and Autonomous Systems. Adam A. Stokes acknowledges support from the EPSRC
32 ORCA Hub (EP/R026173/1).
33
34
35
36

37 Author Disclosure Statement

38
39
40 No competing financial interests exist.
41
42

43 Figure Captions

44
45 **Figure 1: System overview of our adaptive sorter.** A) The assembled adaptive sorting system. The
46 flexible surface is made of spray painted acetate and is attached to the Linbots beneath with Velcro.
47 B) Shows the 5x5 grid of Linbots that control the height of the surface.
48
49

50
51 **Figure 2: Peristaltic wave control.** A) A schematic of a peristaltic wave at one point in time. The
52 wave is formed from the cooperation of multiple Linbots. The wave moves from left to right. At B)
53 The displacement of one Linbot during a wave motion. Axes not to scale. C) Addressing system
54
55
56
57
58
59
60

1
2 based on communication hops. The addressing system allows the robots to know the source of
3
4 commands.

5
6 **Figure 3:** Vibration for destabilisation and conveyance. A) A tomato with highlighted edges showing
7 that it is not a perfect spheroid. The range of contact points that will lead to stable and unstable poses
8 are shown. To move from a tomato out of a stable pose a Linbot rapidly extends and contracts to
9 shake the tomato. B) Hybrid vibrational and peristaltic motion. The ability of the Linbots to rapidly
10 extend and contract can also be used to convey objects that do not slide or roll. The Linbots on either
11 side of a piece of cargo expand or contract to create a gradient in the desired direction of travel. The
12 Linbot beneath the cargo then starts to vibrate. As the Linbot extends, the gradient of the surface
13 biases the motion such that the cargo is pushed both upwards and down the gradient. When the
14 Linbot contracts the object falls directly downwards under gravity. The rapid repetition of these steps
15 while vibrating causes the cargo to move down the gradient. The Linbots change roles as the cargo
16 moves between them. C) Freeze frames from Video S6 showing the results of the vibrational
17 conveyance experiment. A 50mm disk was used that would not roll or slide down the gradients
18 created by the sorter. The disk was successfully moved 100mm in 8s through combined vibrational
19 and peristaltic motion.
20
21
22
23
24
25
26
27
28
29
30

31 **Figure 4: The path planning experiment.** A) An overview of the experiment. The tomato needs to
32 be routed from the top left of the table to the top right but a direct route is blocked. The state grid is
33 shown as well as two possible paths, one optimal and one suboptimal B) An Example of a probability
34 distribution of resultant states for a state-action pair. The brightness of a state shows how likely a
35 resultant state it is. The starting state is (1,3) and the action is a one Linbot long peristaltic wave
36 downwards. The state numbers and cartesian grid coordinates are shown. C) The expected cost of
37 reaching the goal and best action for each state. D) The results of the real-world test showing the
38 planned route and the path taken by the tomato. The path planned by the efficient choice algorithm
39 matches the human selected optimal path in Figure 4A.
40
41
42
43
44
45
46
47

48 **Figure 5: Results of the mass sorting task.** A) The performance of the sorting system. The sorting
49 task is a combination of two sub-tasks: 1) A classification task where the system must decide which
50 class the tomato belongs to. 2) A movement task where the system moves the tomato off of the
51 correct side of the table for the class. The results for the subtasks and the combined tasks are shown,
52 along with the classification matrix. B) The distribution of the errors in both tasks in relation to the
53 mass of the tomato being sorted.
54
55
56
57
58
59
60

1
2 **Figure 6: Freeze frames from the results of the multi-sorting and shape sorting experiments. A)**

3
4 The result of the multi-sorting task, shown by Video S4. 0s: the Linbots try to create two movements
5 that intersect. 4s: the movement towards the left of the table reserves the Linbot at the crossing of the
6 two movements and is executed, the downwards movement waits for it to be completed. 6s: the
7 movement to the left has completed and the Linbots involved have become dormant. 21s: having
8 come out of dormancy the Linbot at the crossing point of the two movements is now reserved to the
9 downwards movement and the movement executes. 22s: both movements have been completed
10 successfully. B) The results of the shape sorting task, shown by video S5. 0s: a less spheroid tomato
11 is placed on the sorter. 5s: having detected an object, the Linbot underneath the tomato tries to move
12 it to the left. The tomatoes shape means that it remains in its stable pose and is not moved. 24s:
13 having detected that a weight remains the Linbot now uses a higher amplitude vibration to destabilise
14 the less spheroid tomato and moves it to the right. 26s: a more spheroid tomato is placed on the
15 sorter. 31s: having detected an object, the Linbot beneath the tomato tries to move it to the left. As
16 the tomato is more spheroid it is easier to destabilise and so it is moved.
17
18
19
20
21
22
23
24
25
26

27 **Figure 7: Freeze frames from the results of the vibrational separation task.** This task is shown
28 by Video S8. At ~3.2s the centre Linbot starts vibrating at 144Hz. These vibrations cause the
29 granular medium (salt) to move down the gradient while not affecting the tomato, leading to
30 separation. The long time between majority separation (~5s) and almost complete separation (~60s)
31 is caused by the mass of the tomato suppressing the vibrations in an area near it.
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60

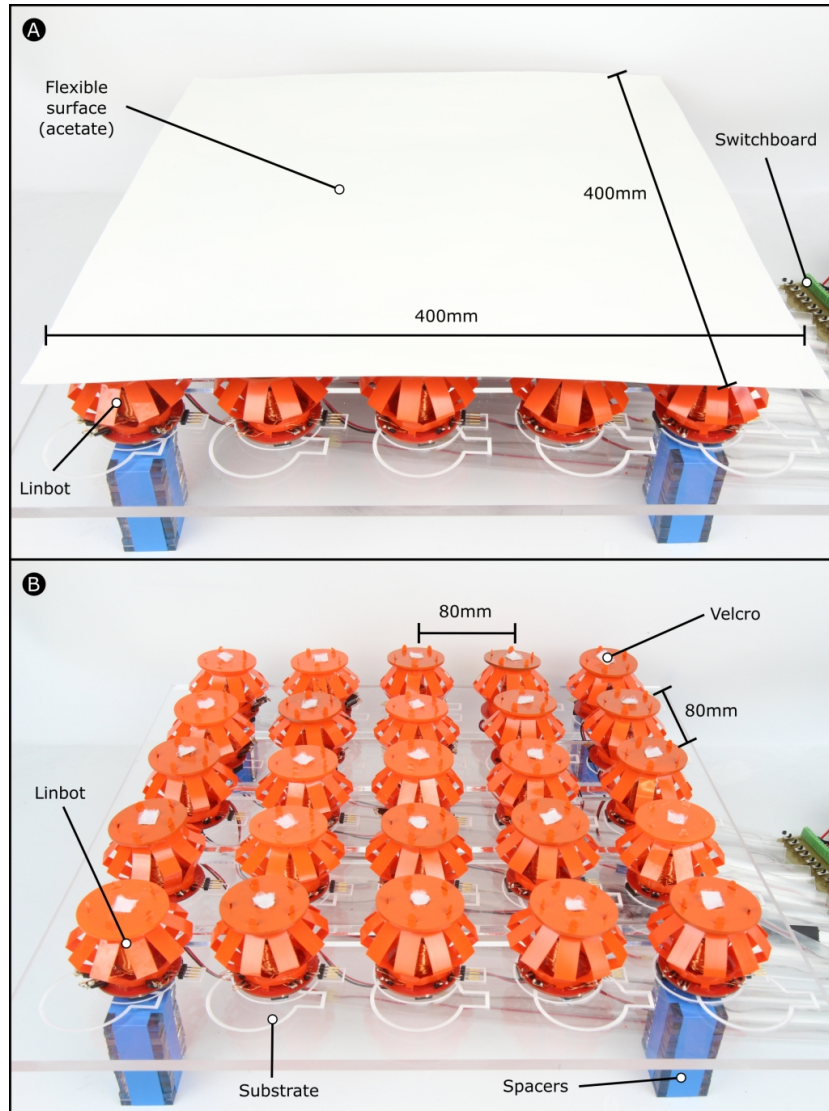


Figure 1: System overview of our adaptive sorter. A) The assembled adaptive sorting system. The flexible surface is made of spray painted acetate and is attached to the Linbots beneath with Velcro. B) Shows the 5x5 grid of Linbots that control the height of the surface.

Mary Ann Liebert, Inc., 140 Huguenot Street, New Rochelle, NY 10801

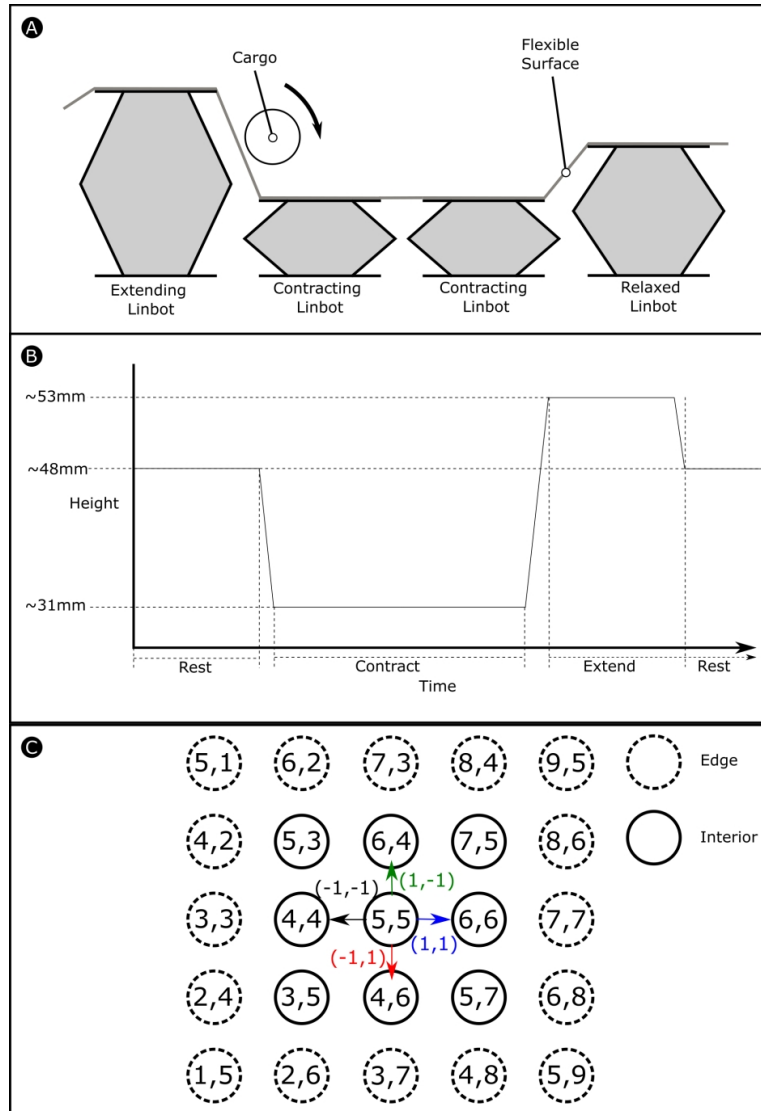


Figure 2: Peristaltic wave control. A) A schematic of a peristaltic wave at one point in time. The wave is formed from the cooperation of multiple Linbots. The wave moves from left to right. At B) The displacement of one Linbot during a wave motion. Axes not to scale. C) Addressing system based on communication hops. The addressing system allows the robots to know the source of commands.

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60

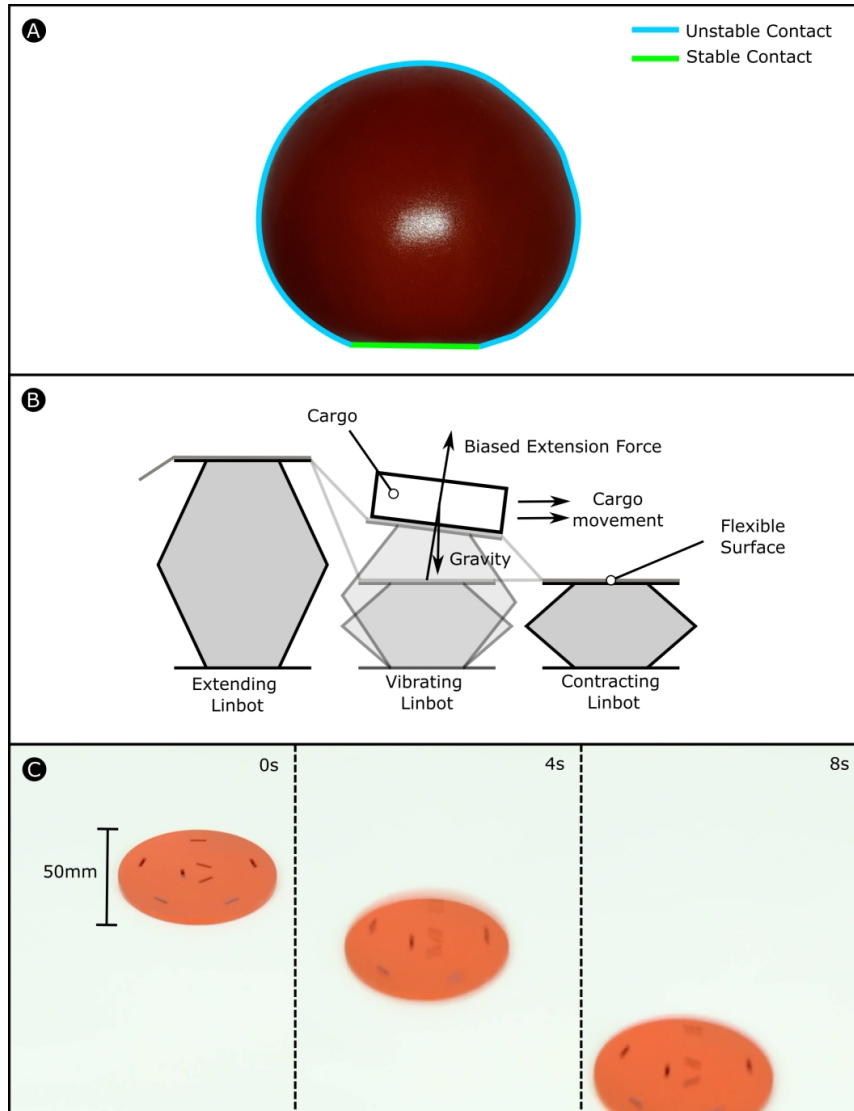


Figure 3: Vibration for destabilisation and conveyance. A) A tomato with highlighted edges showing that it is not a perfect spheroid. The range of contact points that will lead to stable and unstable poses are shown. To move from a tomato out of a stable pose a Linbot rapidly extends and contracts to shake the tomato. B) Hybrid vibrational and peristaltic motion. The ability of the Linbots to rapidly extend and contract can also be used to convey objects that do not slide or roll. The Linbots on either side of a piece of cargo expand or contract to create a gradient in the desired direction of travel. The Linbot beneath the cargo then starts to vibrate. As the Linbot extends, the gradient of the surface biases the motion such that the cargo is pushed both upwards and down the gradient. When the Linbot contracts the object falls directly downwards under gravity. The rapid repetition of these steps while vibrating causes the cargo to move down the gradient. The Linbots change roles as the cargo moves between them. C) Freeze frames from Video S6 showing the results of the vibrational conveyance experiment. A 50mm disk was used that would not roll or slide down the gradients created by the sorter. The disk was successfully moved 100mm in 8s through combined vibrational and peristaltic motion.

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60

Mary Ann Liebert, Inc., 140 Huguenot Street, New Rochelle, NY 10801

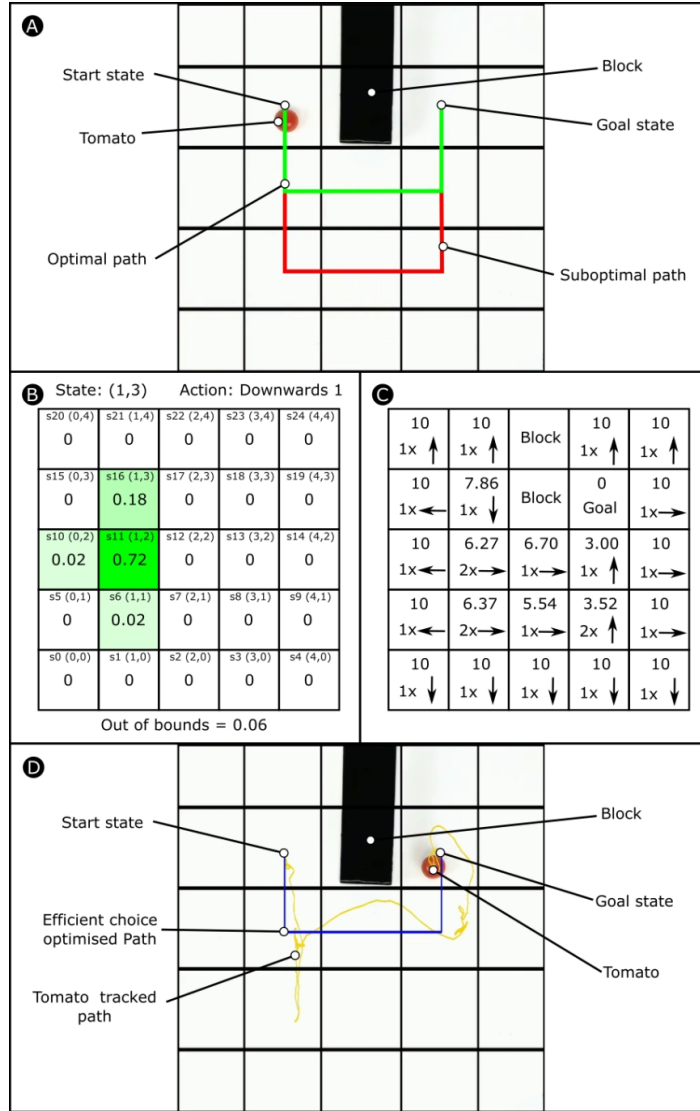


Figure 4: The path planning experiment. A) An overview of the experiment. The tomato needs to be routed from the top left of the table to the top right but a direct route is blocked. The state grid is shown as well as two possible paths, one optimal and one suboptimal B) An Example of a probability distribution of resultant states for a state-action pair. The brightness of a state shows how likely a resultant state it is. The starting state is (1,3) and the action is a one Linbot long peristaltic wave downwards. The state numbers and cartesian grid coordinates are shown. C) The expected cost of reaching the goal and best action for each state. D) The results of the real-world test showing the planned route and the path taken by the tomato. The path planned by the efficient choice algorithm matches the human selected optimal path in Figure 4A.

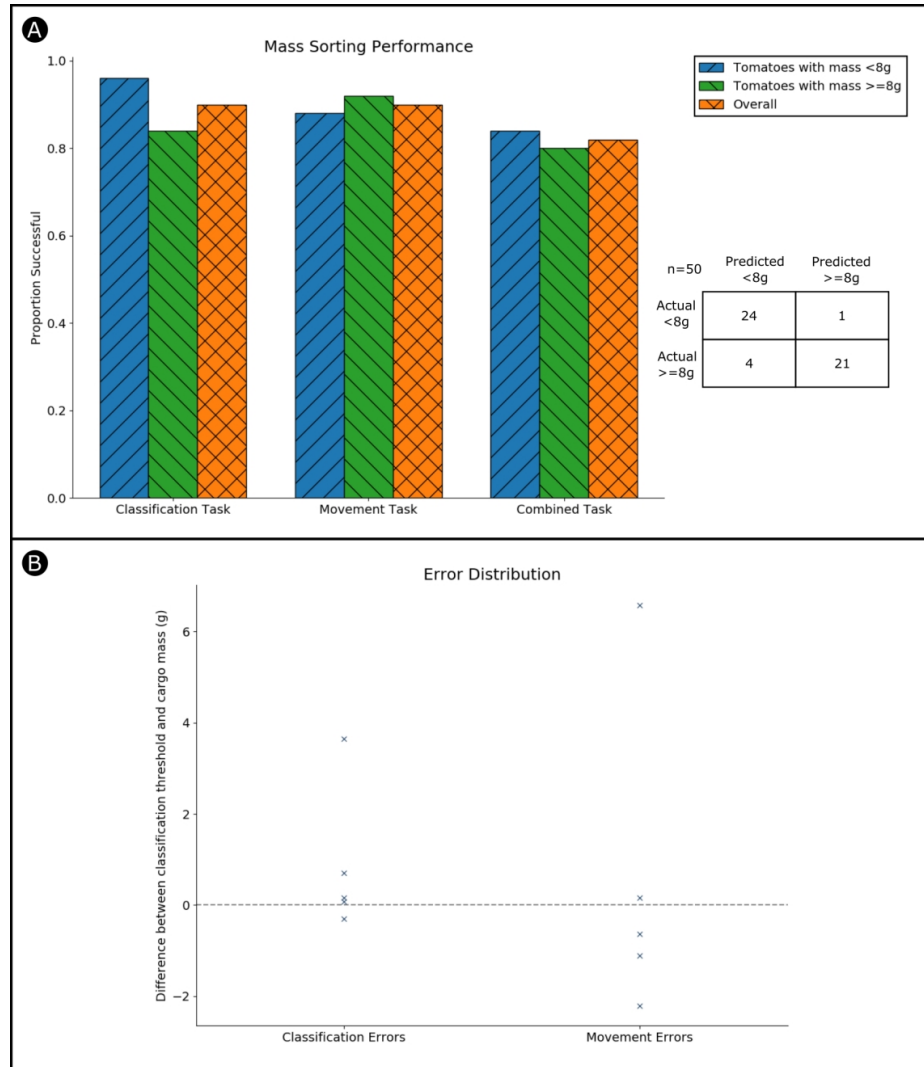


Figure 5: Results of the mass sorting task. A) The performance of the sorting system. The sorting task is a combination of two sub-tasks: 1) A classification task where the system must decide which class the tomato belongs to. 2) A movement task where the system moves the tomato off of the correct side of the table for the class. The results for the subtasks and the combined tasks are shown, along with the classification matrix. B) The distribution of the errors in both tasks in relation to the mass of the tomato being sorted.

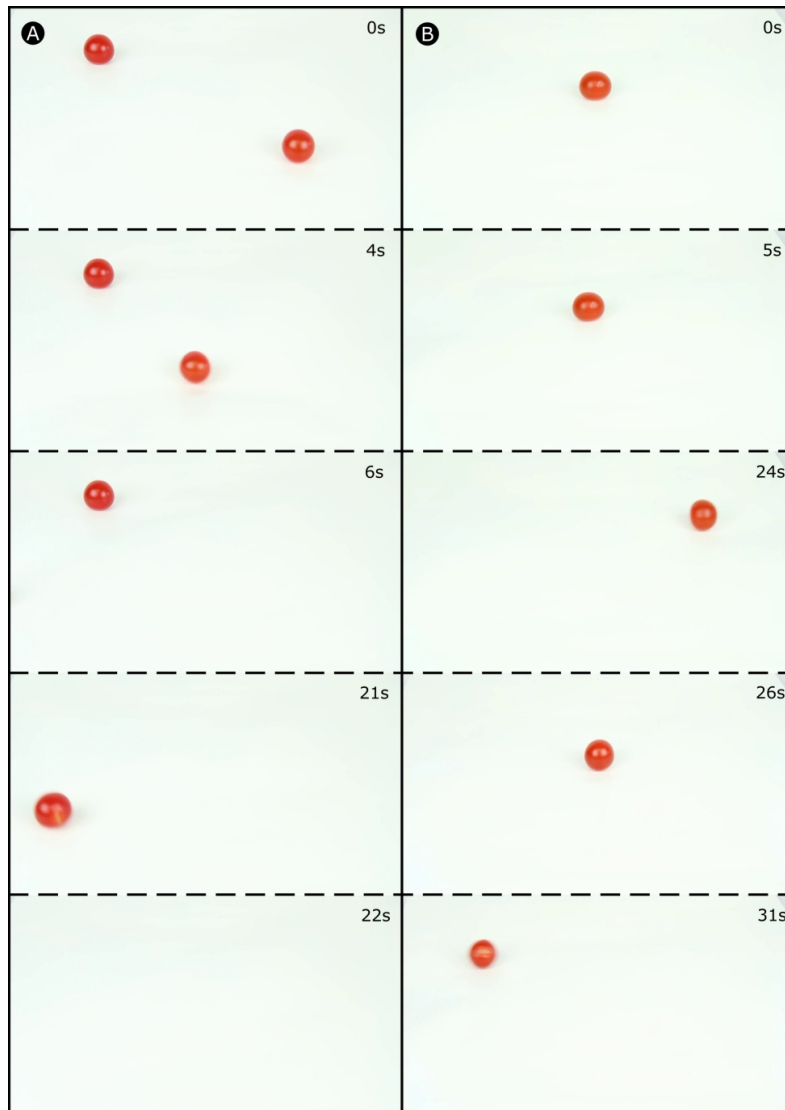


Figure 6: Freeze frames from the results of the multi-sorting and shape sorting experiments. A) The result of the multi-sorting task, shown by Video S4. 0s: the Linbots try to create two movements that intersect. 4s: the movement towards the left of the table reserves the Linbot at the crossing of the two movements and is executed, the downwards movement waits for it to be completed. 6s: the movement to the left has completed and the Linbots involved have become dormant. 21s: having come out of dormancy the Linbot at the crossing point of the two movements is now reserved to the downwards movement and the movement executes. 22s: both movements have been completed successfully. B) The results of the shape sorting task, shown by video S5. 0s: a less spheroid tomato is placed on the sorter. 5s: having detected an object, the Linbot underneath the tomato tries to move it to the left. The tomatoes shape means that it remains in its stable pose and is not moved. 24s: having detected that a weight remains the Linbot now uses a higher amplitude vibration to destabilise the less spheroid tomato and moves it to the right. 26s: a more spheroid tomato is placed on the sorter. 31s: having detected an object, the Linbot beneath the tomato tries to move it to the left. As the tomato is more spheroid it is easier to destabilise and so it is moved.

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60

Mary Ann Liebert, Inc., 140 Huguenot Street, New Rochelle, NY 10801

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60

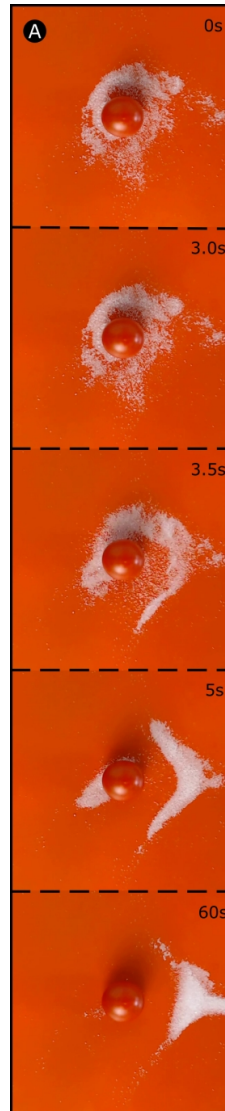
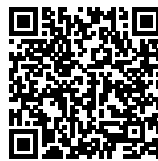


Figure 7: Freeze frames from the results of the vibrational separation task. This task is shown by Video S8. At ~ 3.2 s the centre Linbot starts vibrating at 144Hz. These vibrations cause the granular medium (salt) to move down the gradient while not affecting the tomato, leading to separation. The long time between majority separation (~ 5 s) and almost complete separation (~ 60 s) is caused by the mass of the tomato suppressing the vibrations in an area near it.

To further clarify the contributions of M. Sayed: I was the project lead and so made the final decisions on all aspects of the project. M. Sayed had input on all aspects of the project. I performed most of the algorithm design and coding with M. Sayed as a reviewer. M. Sayed planned the circuit board layout based on a schematic created by both of us, I edited the layout. I planned the structure of the paper. M. Sayed drafted the supplemental information. I drafted the main paper. I built the coil winder controller. All other tasks involved were done by both of us working together and cannot be clearly separated into sub-tasks done by one of us.

5.3 Supplemental Videos

For the supplemental videos for this publication scan this QR code:



5.4 Conclusion

This chapter detailed my construction and control of a modular robotic, adaptive sorter. I have shown that modular robots can be used to build a sorting system with flexibility, that is able to handle a variety of objects and sort for more than one criteria. The sorting system suffers from inconsistencies in construction due to being a hand made prototype. Having a very level surface and more consistent actuation behaviour would remove most of the errors in sorting and path planning detailed in this behaviour. Some errors, however, are inherent to working with real world objects such as those caused by non-spherical objects not rolling in repeatable ways. In this work I built a system robust against some of these errors.

The system represents a starting point for a novel sorting system specifically designed to work well on varieties of objects or in rapidly changing conditions. As robotics expands into less structured environments more adaptive systems will be required to fulfil the demand for autonomous cargo sorting.

Chapter 6

Discussion and future research

My thesis aimed to find a middle ground between the flexible pick and place systems common in sorting research and the high throughput and low cost sorters common throughout industry. To achieve that end I identified a promising sorting system design in peristaltic sorters and identified major problems with existing implementations. I developed modular robots called Linbots that have the potential to allow for distributed peristaltic systems. I also developed a peristaltic table simulator, PeriSim, to aid with control design as there was a lack of available simulators. I built an adaptive sorting system from my Linbots to demonstrate the use of a peristaltic table as an adaptive sorting system.

6.1 Future Developments

6.1.1 Integration with Additional Sensors

To make the adaptive sorting system even more flexible extra sensors could be added to the Linbots. The Linbot design includes I²C pins for the attachment of such extra sensors. Some examples of extra sensors could be:

- A temperature sensor in contact with the surface above the Linbot to monitor the state of any cargo.
- Additional magnetic sensors for determining the cargo composition
- Sound sensors for detecting cargo collisions or for acoustic communication

Additionally, further research into using the Linbots in sensor arrays or as distributed sensors could both find new applications for the Linbots and develop sensing algorithms for Linbot-based sorting systems.

6.1.2 Advanced Vibrational Sorting

With further mechanical refinement vibrational control using the Linbot system could be much improved. The ability to sync up robot movements could allow for collective-wide vibration mimicking a large vibrational conveyor and allowing large throughput transport. High enough frequency vibration can produce standing waves in the surface of the sorter. Small particles move into the nodes of these waves producing Chladni

patterns [57], control of these patterns can allow for the control of small objects at a higher resolution than is possible with the current system.

6.1.3 Integration of Central Advisor Systems

Some sensor systems, such as cameras, can be better used as central single systems rather than deployed on all of the robots in a collective. These systems come with a trade-off of being not inherently scalable and malfunctions causing more disruption. The Linbots are capable of disseminating information through a collective, this functionality creates the potential for outside systems to provide collective-wide sensor information or commands. Global systems advising intelligent collectives have the potential to provide the best aspects of both centralised and decentralised control. For example a camera above a Linbot collective could provide visual information to a collective allowing the adaptive sorting system to sort by new criteria without needing to add visual sensors to all of the robots. Central control can also allow the system to quickly respond to global changes such as needing to sweep the sorting area clear due to a malfunction or incoming cargo change. These central advisor systems can also allow the user other avenues of control outside of the current use of the Linbots as push buttons.

6.1.4 Online Learning

The peristaltic path planning I use in this work runs in a distributed manner but requires a central system for the initial learning of behaviour. This requirement could lead to delays in the system adapting to a new situation and reduces the scalability of the approach. The Linbots have enough computational power to handle some online learning. This learning could take the form of per-robot reinforcement learning with rewards for successful sorts being disseminated across the collective and being collected by robots that took actions that were related to it. This approach could allow for adapting behaviour in the case of new blockages or failures. The system would likely need to be first optimised in simulation and the robots given initial policies to avoid an unreasonable amount of time being required for setup, but when running the robots can then adjust their behaviour in a fully distributed way.

6.1.5 3D Collectives

By building more complex 3D collectives the adaptive sorter can be used for a wider variety of situations. A sorting table made of linbots can conform to non-flat surfaces allowing for inbuilt channels and bowls to collect and direct objects. The Linbots can also be stacked in different poses allowing for them to actuate in the XY plane and to push objects directly rather than relying on objects moving down gradients. Linbots arranged into mobile collectives could sort objects internally while also conveying them across a larger area.

6.1.6 Mixed Stream Sorting Application

In order to move my sorting system toward real-world usage further development in a more realistic setting is required. A realistic input stream of mixed recycling could be

used for test cargo while extra sensors attached in a distributed or centralised manner can provide the extra data needed to sort objects not just by size and shape. A larger machine (around 20x20 Linbots) would be required to sort objects on the scale of a recycling centre. A support robotic arm could be used to pick and place any anomalous or stuck objects that the sorting table cannot move or separate from other objects.

6.2 Conclusion

In my thesis I aimed to work in an underexplored area. While most new sorting research looks at individual item movement with pick and place arms or mobile robots I wanted to develop a flexible system that took ideas from the industrial sorting methods we see in use today. Methods like rotating sieves or shaker tables that accomplish sorting tasks for very low cost with no computation. By taking some of the ideas behind mechanical sorting and combining it with robotics I felt I could contribute a new approach to the problem of sorting. Soft robotics was a good fit for my project as it mirrors the themes of using the physical characteristics of materials to control behaviour without computer input. Peristaltic tables, while actively researched, have not yet found wide industrial use due to a lack of focus on their flexibility. Most peristaltic research is focused on refining object rolling behaviour with external sensors. I aimed to build a peristaltic table dedicated to exploring the possibilities in the base design.

I have designed and demonstrated a novel, soft, modular robot, the Linbot. This robot can function as a cell in a peristaltic system allowing the system to use distributed algorithms. Distributing the computation of the system allows any peristaltic table made from the Linbots to be both scalable and parallel if given a scalable control algorithm. The Linbot is not limited to this role, it's design represents a basic extension and contraction function found in animal muscles and so the Linbots could be used as modular muscle components in other robotic collectives. I designed the Linbot for manufacturability, and so it is cheap and easy to construct.

I have written a peristaltic table simulator that I have published as the python package PeriSim. This simulator can be tuned to different systems and has the potential to be used for other peristaltic table research. Previously there was a lack of available peristaltic table simulations, adding a barrier to developing peristaltic control in simulation. My simulator is designed to be easy to use and is available for other researchers. It overcomes the difficulties with modelling peristaltic behaviour using a radical envelope of noise that allows for controllers designed in a simulation to be robust to the changes in dynamics when moved to the real-world.

I created a state-machine based algorithm for low-level peristaltic wave control. This algorithm allows for controlling the speed, direction, and length of a peristaltic wave in a distributed manner. This algorithm also avoids collisions between objects moving on a peristaltic table. The controller can be used for both distributed and centralised systems. Without the need to redesign low-level peristaltic control for each new peristaltic table the barrier to entry for peristaltic table researchers is reduced.

I have constructed and demonstrated a proof of concept, modular robotic, adaptive

sorting system. I have demonstrated the potential for future iterations of the system to sort heterogeneous streams of objects. My system can serve as a proof of concept for scalable, distributed sorting systems. A larger, more developed version of my system can be combined with diverse sensor systems to sort heterogeneous object streams found in industries such as recycling and port authorities.

I have been lucky to work on a project with so much freedom to explore new ideas and I enjoyed contributing to this fascinating field.

Bibliography

- [1] S. P. Gundupalli, S. Hait, and A. Thakur, “A review on automated sorting of source-separated municipal solid waste for recycling,” *Waste Management*, vol. 60, pp. 56 – 74, 2017. Special Thematic Issue: Urban Mining and Circular Economy.
- [2] S. Pellegrinelli, “Configuration and reconfiguration of robotic systems for waste macro sorting,” *The International Journal of Advanced Manufacturing Technology*, vol. 102, pp. 3677–3687, June 2019.
- [3] A. Zeng, S. Song, K. Yu, E. Donlon, F. R. Hogan, M. Bauza, D. Ma, O. Taylor, M. Liu, E. Romo, N. Fazeli, F. Alet, N. C. Daffe, R. Holladay, I. Morena, P. Qu Nair, D. Green, I. Taylor, W. Liu, T. Funkhouser, and A. Rodriguez, “Robotic pick-and-place of novel objects in clutter with multi-affordance grasping and cross-domain image matching,” in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1–8, May 2018.
- [4] O. Technology, “Ocado technology home page,” 2019.
- [5] C. Liang, K. Chee, Y. Zou, H. Zhu, A. Causo, S. Vidas, T. Teng, I. Chen, K. Low, and C. Cheah, “Automated robot picking system for e-commerce fulfillment warehouse application,” The 14th IFToMM World Congress, Taipei, Taiwan, October 25-30, 2015, 2015.
- [6] R. Pellenc and J.-M. Gialis, “Sorting table with sorter rolls for elimination of foreign matter remaining mixed in a harvest of small fruit,” 2007.
- [7] HM Revenue Customs, “Classifying rice for import and export - gov.uk,” 2015.
- [8] P. Chen and Z. Sun, “A review of non-destructive methods for quality evaluation and sorting of agricultural products,” *Journal of Agricultural Engineering Research*, vol. 49, no. Supplement C, pp. 85 – 98, 1991.
- [9] S. Rokade, S. Tapare, and C. Pawar, “A parallel study of designing and simulation of industrial robotics,” *International Journal of Management and Applied Science (IJMAS)*, vol. 5, no. 2, pp. 55–60, 2017.
- [10] S. Gundupalli Paulraj, S. Hait, and A. Thakur, “Automated municipal solid waste sorting for recycling using a mobile manipulator,” 2016.
- [11] M. Stommel, W. L. Xu, P. P. K. Lim, and B. Kadmiry, *Soft Peristaltic Actuation for the Harvesting of Ovine Offal*, pp. 605–615. Cham: Springer International Publishing, 2015.
- [12] T. E. of Encyclopaedia Britannica, “Peristalsis,” *Encyclopædia Britannica*, 2019.

-
- [13] Y. Yamada, K. Ashigaki, S. Yoshihama, K. Negishi, K. Kato, and T. Nakamura, “Triangular cross-section peristaltic conveyor for transporting powders at high speed in printers,” *Advanced Robotics*, vol. 32, no. 12, pp. 646–658, 2018.
- [14] M. Stommel and W. Xu, “Optimal, efficient sequential control of a soft-bodied, peristaltic sorting table,” *IEEE Transactions on Automation Science and Engineering*, vol. 13, pp. 858–867, April 2016.
- [15] R. M. McKenzie, M. E. Sayed, M. P. Nemitz, B. W. Flynn, and A. A. Stokes, “Linbots: Soft modular robots utilising voice coils,” *Soft Robotics*, vol. 6, no. 2, pp. 195–205, 2019.
- [16] R. M. McKenzie, J. O. Roberts, M. E. Sayed, and A. A. Stokes, “Perisim: A simulator for optimising peristaltic table control,” *Advanced Intelligent Systems*, vol. 1, no. 8, p. 1900070, 2019.
- [17] R. M. McKenzie, J. O. Roberts, M. E. Sayed, and A. A. Stokes, “A modular robotic sorting table,” *Soft Robotics*, In Review.
- [18] R. Hashem, B. Smith, D. Browne, W. Xu, and M. Stommel, “Control of a soft-bodied xy peristaltic table for delicate sorting,” in *2016 IEEE 14th International Workshop on Advanced Motion Control (AMC)*, pp. 358–363, April 2016.
- [19] T. Tone and K. Suzuki, “An automated liquid manipulation by using a ferrofluid-based robotic sheet,” *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 2814–2821, 2018.
- [20] Z. Deng, M. Stommel, and W. Xu, “A novel soft machine table for manipulation of delicate objects inspired by caterpillar locomotion,” *IEEE/ASME Transactions on Mechatronics*, vol. 21, pp. 1702–1710, June 2016.
- [21] S.-R. Kim, D.-Y. Lee, J.-S. Koh, and K.-J. Cho, “Fast, compact, and lightweight shape-shifting system composed of distributed self-folding origami modules,” in *ICRA* (D. Kragic, A. Bicchi, and A. D. Luca, eds.), pp. 4969–4974, IEEE, 2016.
- [22] N. B. Cruz, N. Nedjah, and L. de Macedo Mourelle, *Efficient Spacial Clustering in Swarm Robotics*, pp. 14–27. Cham: Springer International Publishing, 2015.
- [23] H. Ding and H. Hamann, *Sorting in Swarm Robots Using Communication-Based Cluster Size Estimation*, pp. 262–269. Cham: Springer International Publishing, 2014.
- [24] J. Chen, M. Gauci, and R. Groß, “A strategy for transporting tall objects with a swarm of miniature mobile robots,” in *2013 IEEE International Conference on Robotics and Automation*, pp. 863–869, May 2013.
- [25] A. Sproewitz, A. Billard, P. Dillenbourg, and A. J. Ijspeert, “Roombots-mechanical design of self-reconfiguring modular robots for adaptive furniture,” in *2009 IEEE International Conference on Robotics and Automation*, pp. 4259–4264, May 2009.

-
- [26] K. Nakagaki, A. Dementyev, S. Follmer, J. A. Paradiso, and H. Ishii, “Chainform: A linear integrated modular hardware system for shape changing interfaces,” in *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, UIST '16, (New York, NY, USA), pp. 87–96, ACM, 2016.
- [27] R. Moeckel, C. Jaquier, K. Drapel, E. Dittrich, A. Upegui, and A. J. Ijspeert, “Exploring adaptive locomotion with YaMoR, a novel autonomous modular robot with bluetooth interface,” *Industrial Robot: An International Journal*, vol. 33, pp. 285–290, July 2006.
- [28] V. Zykov, A. Chan, and H. Lipson, “Molecubes: An open-source modular robotic kit,” in *in IROS-2007 Self-Reconfigurable Robotics Workshop*, 2007.
- [29] J. W. Romanishin, K. Gilpin, and D. Rus, “M-blocks: Momentum-driven, magnetic modular robots,” in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4288–4295, Nov. 2013.
- [30] 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, pp. 1–17, 01 2017.
- [31] D. Zappetti, S. Mintchev, J. Shintake, and D. Floreano, *Bio-inspired Tensegrity Soft Modular Robots*, pp. 497–508. Cham: Springer International Publishing, 2017.
- [32] M. A. Robertson and J. Paik, “New soft robots really suck: Vacuum-powered systems empower diverse capabilities,” *Science Robotics*, vol. 2, p. eaan6357, Aug. 2017.
- [33] M. P. Nemitz, P. Mihaylov, T. W. Barraclough, D. Ross, and A. A. Stokes, “Using voice coils to actuate modular soft robots: Wormbot, an example,” *Soft Robotics*, vol. 3, no. 4, pp. 198–204, 2016.
- [34] L. Panait and S. Luke, “Cooperative multi-agent learning: The state of the art,” *Autonomous Agents and Multi-Agent Systems*, vol. 11, pp. 387–434, Nov. 2005.
- [35] W. Liu and A. F. Winfield, “Modelling and optimisation of adaptive foraging in swarm robotic systems,” *The International Journal of Robotics Research*, vol. 29, pp. 1743–1760, December 2010.
- [36] D. J. Christensen, U. P. Schultz, and K. Stoy, “A distributed and morphology-independent strategy for adaptive locomotion in self-reconfigurable modular robots,” *Robotics and Autonomous Systems*, vol. 61, no. 9, pp. 1021 – 1035, 2013.
- [37] S. Jones, A. F. Winfield, S. Hauert, and M. Studley, “Onboard evolution of understandable swarm behaviors,” *Advanced Intelligent Systems*, vol. 1, no. 6, p. 1900031, 2019.
- [38] H. Guo and Y. Meng, “Distributed reinforcement learning for coordinate multi-robot foraging,” *Journal of Intelligent & Robotic Systems*, vol. 60, pp. 531–551, Dec. 2010.
- [39] D. Sudholt, *Parallel Evolutionary Algorithms*, pp. 929–959. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015.

-
- [40] A. Nair, P. Srinivasan, S. Blackwell, C. Alcicek, R. Fearon, A. De Maria, V. Panneershelvam, M. Suleyman, C. Beattie, S. Petersen, S. Legg, V. Mnih, K. Kavukcuoglu, and D. Silver, “Massively parallel methods for deep reinforcement learning,” 2015. cite arxiv:1507.04296Comment: Presented at the Deep Learning Workshop, International Conference on Machine Learning, Lille, France, 2015.
- [41] D. Whitley, “A genetic algorithm tutorial,” *Statistics and Computing*, vol. 4, pp. 65–85, June 1994.
- [42] C.-U. Lim, R. Baumgarten, and S. Colton, *Evolving Behaviour Trees for the Commercial Game DEFCON*, pp. 100–110. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010.
- [43] S. Jones, M. Studley, S. Hauert, and A. Winfield, *Evolving behaviour trees for swarm robotics*. Springer Tracts in Advanced Robotics, Springer, 9 2016.
- [44] H. Hamann, J. Stradner, T. Schmickl, and K. Crailsheim, “A hormone-based controller for evolutionary multi-modular robotics: From single modules to gait learning,” in *IEEE Congress on Evolutionary Computation*, pp. 1–8, July 2010.
- [45] M. Stommel, Z. Deng, and W. Xu, “Probabilistic automata model of a soft robot for the planning of manipulation tasks,” *IEEE Trans. Automation Science and Engineering*, vol. 14, no. 4, pp. 1722–1730, 2017.
- [46] K. Scheper, “Behaviour trees for evolutionary robotics: Reducing the reality gap,” Master’s thesis, 2014.
- [47] P. Abbeel, M. Quigley, and A. Y. Ng, “Using inaccurate models in reinforcement learning,” in *Proceedings of the 23rd International Conference on Machine Learning, ICML ’06*, (New York, NY, USA), pp. 1–8, ACM, 2006.
- [48] A. Nagabandi, G. Yang, T. Asmar, G. Kahn, S. Levine, and R. S. Fearing, “Neural network dynamics models for control of under-actuated legged millirobots,” 2017. cite arxiv:1711.05253.
- [49] N. Jakobi, “Evolutionary robotics and the radical envelope-of-noise hypothesis,” *Adaptive Behavior*, vol. 6, no. 2, pp. 325–368, 1997.
- [50] M. Wang, “Rlc circuits and resonance,” 2010.
- [51] T. Paulino, P. C. Ribeiro, M. Neto, S. Cardoso, A. Schmitz, J. Santos-Victor, A. Bernardino, and L. Jamone, “Low-cost 3-axis soft tactile sensors for the human-friendly robot vizzy,” in *ICRA*, pp. 966–971, IEEE, 2017.
- [52] N. Koenig and A. Howard, “Design and use paradigms for gazebo, an open-source multi-robot simulator,” in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, vol. 3, pp. 2149–2154 vol.3, Sep. 2004.
- [53] U. Technologies, “Unity engine manual,” 2019. accessed: 26/09/2019.
- [54] S. Consortium, “Sofa,” 2007. Accessed: 17/06/2017.

-
- [55] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, “Automatic differentiation in pytorch,” in *NIPS 2017 Workshop on Autodiff*, 2017.
- [56] P. P. Authority, “pip,” 2019. accessed: 26/09/2019.
- [57] Q. Zhou, V. Sariola, K. Latifi, and V. Liimatainen, “Controlling the motion of multiple objects on a chladni plate,” *Nature Communications*, vol. 7, no. 1, pp. 12764–, 2016.
- [58] E. Hegazi, J. Rael, and A. Abidi, *Basics of LC Oscillators*, pp. 1–9. Boston, MA: Springer US, 2005.
- [59] G. Tech, “Darlington pair,” 2013. Accessed: 12/07/2017.
- [60] U. St. Andrews, “The envelope detector,” 2000. Accessed: 15/06/2017.
- [61] J. R. Khan, “Zener diode basic operation and applications,” 2015. Accessed: 15/06/2017.
- [62] D. O. Pederson and K. Mayaram, *Analog integrated circuits for communication : principles, simulation and design*. Berkeley, CA: Springer, 2008.

Appendices

Appendix A

2D Designs

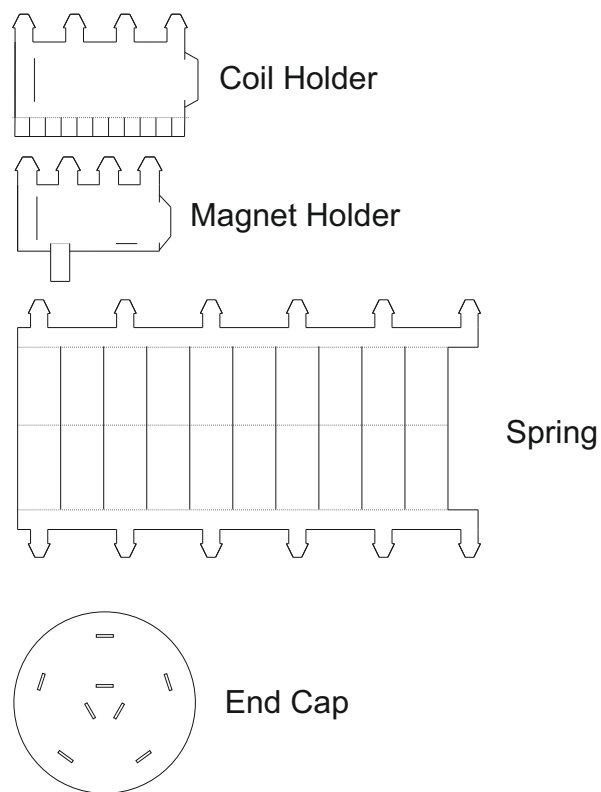


Figure A.1: 2D designs for Linbot components

Appendix B

Bill of Materials

Part Number	Description	Value	Manufacturer	Manufacturer Part Number	Cost(£)	Supplier	Supplier Part Number
Communication (Transmission Circuit)							
R1	0402 Resistor	10k	Bourns	CR0402-FX-1002GLF	0.025	RS Components	788-3990
R2	0402 Resistor	680	Vishay	CRCW0402680RFKED	0.013	RS Components	678-9510
C1	0402 Capacitor	1p	Phycomp	223886915108	0.003	RS Components	456-8566
C2	0402 Capacitor	1u	TDK	C1005X5R0J105K050BB	0.008	RS Components	741-4399
C3	0402 Capacitor	10n	KEMET	C0402C103J3RACTU	0.017	RS Components	801-5151
Q2	NPN Transistor	N/A	NXP	BC817-25	0.008	RS Components	436-7903
Q5	N-channel Mosfet	N/A	Fairchild Semiconductor	FQB30N06LTM	0.388	RS Components	671-0882
Communication (Receiver Circuit)							
R3	0402 Resistor	10k	Bourns	CR0402-FX-1002GLF	0.025	RS Components	788-3990
R4	0402 Resistor	4.7K	TE Connectivity	CRG0402J4K7/10	0.002	RS Components	371-5086
R5	0402 Resistor	4.3K	Vishay	CRCW04024K30FKED	0.002	RS Components	678-9386
R6	0402 Resistor	5.6K	Vishay	CRCW04025K60FKED	0.013	RS Components	678-9475
R7	0402 Resistor	10k	Bourns	CR0402-FX-1002GLF	0.025	RS Components	788-3990
R8	0402 Resistor	4.7K	TE Connectivity	CRG0402J4K7/10	0.002	RS Components	371-5086
R9	0402 Resistor	3.4K	Vishay	CRCW04023K48FKED	0.013	RS Components	678-9279
R10	0402 Resistor	10k	Bourns	CR0402-FX-1002GLF	0.025	RS Components	788-3990
R11	0402 Resistor	1M	Vishay	CRCW04021M00FKED	0.013	RS Components	678-8923
R12	0402 Resistor	1M	Vishay	CRCW04021M00FKED	0.013	RS Components	678-8923
R13	0402 Resistor	10k	Bourns	CR0402-FX-1002GLF	0.025	RS Components	788-3990
R14	0402 Resistor	330	Vishay	CRCW0402330RFKED	0.002	RS Components	678-9178
R20	0402 Resistor	100	Vishay	CRCW0402100RFKEDHP	0.022	RS Components	812-1568
C4	0402 Capacitor	10p	KEMET	C0402C100J5GACTU	0.007	RS Components	147-926
C5	0402 Capacitor	10n	KEMET	C0402C103J3RACTU	0.017	RS Components	801-5151
C6	0402 Capacitor	10n	KEMET	C0402C103J3RACTU	0.017	RS Components	801-5151
C7	0402 Capacitor	10n	KEMET	C0402C103J3RACTU	0.017	RS Components	801-5151
C8	0402 Capacitor	2.2n	KEMET	C0402C222K5RACTU	0.004	RS Components	147-027
Q3	NPN Transistor	N/A	ON Semiconductor	BCW66GLT3G	0.015	RS Components	781-3976
Q4	NPN Transistor	N/A	ON Semiconductor	BCW66GLT3G	0.015	RS Components	781-3976
U3	Dual Op Amp	N/A	Fairchild Semiconductor	KA358ADTF	0.089	RS Components	807-8770
U4	SMT Schottky Diode	N/A	DiodesZetex	1N6263W-7-F	0.128	RS Components	751-2743
Computation							
U1	Microcontroller	N/A	STMicroelectronics	STM8S003F3U6TR	0.506	RS Components	795-6907
C9	0402 Capacitor	10n	KEMET	C0402C103J3RACTU	0.017	RS Components	801-5151
C14	0402 Capacitor	470p	Murata	GRM155R11H471KA01D	0.004	RS Components	798-0177
User I/O							
LED1	Green LED (0603)	N/A	Osram Opto Semiconductors	LP L296-J2L2-25	0.09	RS Components	915-4768
LED2	Blue LED (0603)	N/A	Vishay	VLM81310-GS08	0.035	RS Components	773-0449
LED3	Yellow LED (0603)	N/A	Lite-on	LTSF-C191KSKT	0.062	RS Components	692-1017
R15	0402 Resistor	330	Vishay	CRCW0402330RFKED	0.002	RS Components	678-9178
R16	0402 Resistor	330	Vishay	CRCW0402330RFKED	0.002	RS Components	678-9178
R17	0402 Resistor	330	Vishay	CRCW0402330RFKED	0.002	RS Components	678-9178
Connectors							
J1	JTAG Connector	4 pins	Samtec	TSM-104-02-L-SH	0.63	RS Components	765-6499
J2	Coil Connectors	2 pins	RS PRO	W81102T3825RC	0.054	RS Components	251-8086
J3	Coil Connectors	2 pins	RS PRO	W81102T3825RC	0.054	RS Components	251-8086
J4	Coil Connectors	2 pins	RS PRO	W81102T3825RC	0.054	RS Components	251-8086
J5	Coil Connectors	2 pins	RS PRO	W81102T3825RC	0.054	RS Components	251-8086
U9	Battery Connector	2 pins	JST Sales America	S2B-PH-SM4-TB(LF)(SN)	0.427	Digi-Key Electronics	455-1749-1-ND
Power							
U8	Voltage Regulator	3.3V Output	DiodesZetex	ZMR330FTA	0.28	RS Components	710-6313
C10	0402 Capacitor	1u	TDK	C1005X5R0J105K050BB	0.008	RS Components	741-4399
C15	0402 Capacitor	10n	KEMET	C0402C103J3RACTU	0.017	RS Components	801-5151
C16	0402 Capacitor	0.1uF	Murata	GRM155R71C104KA88D	0.003	RS Components	723-5228
Sensing							
U2	Hall-effect Sensor	N/A	Melexis	MLX90393SLW-ABA-011-SP	1.33	Mouser Electronics	482-90393SLWABA011SP
R18	0402 Resistor	5.1K	Vishay	CRCW04025K10FKED	0.013	RS Components	678-9478
R19	0402 Resistor	5.1K	Vishay	CRCW04025K10FKED	0.013	RS Components	678-9478
Actuation							
U7	H-Bridge Driver	N/A	Texas Instruments	DRV8837DSGT	0.973	RS Components	813-6497
R21	Power Resistor	8.2	TE Connectivity	SMW38R2JT	0.176	RS Components	647-6144
R22	Power Resistor	8.2	TE Connectivity	SMW38R2JT	0.176	RS Components	647-6144
Other Components							
N/A	Battery	7.4V	LeGow	N/A	8.19	Amazon UK	N/A
N/A	Crimps	N/A	JST (JAPAN SOLDERLESS TERMINALS)	SPH-002T-PO.5S	0.016	Farnell Element14	1671245
N/A	Connector Housing	N/A	JST (JAPAN SOLDERLESS TERMINALS)	PHR-2	0.0402	Farnell Element14	3616186
N/A	PCB Board	N/A	Ragworm	N/A	7.7	Ragworm	N/A

Figure B.1: Bill of Materials. Components cost £21.88 for each Linbot when buying enough for 10 Linbots

Appendix C

Electronic Design

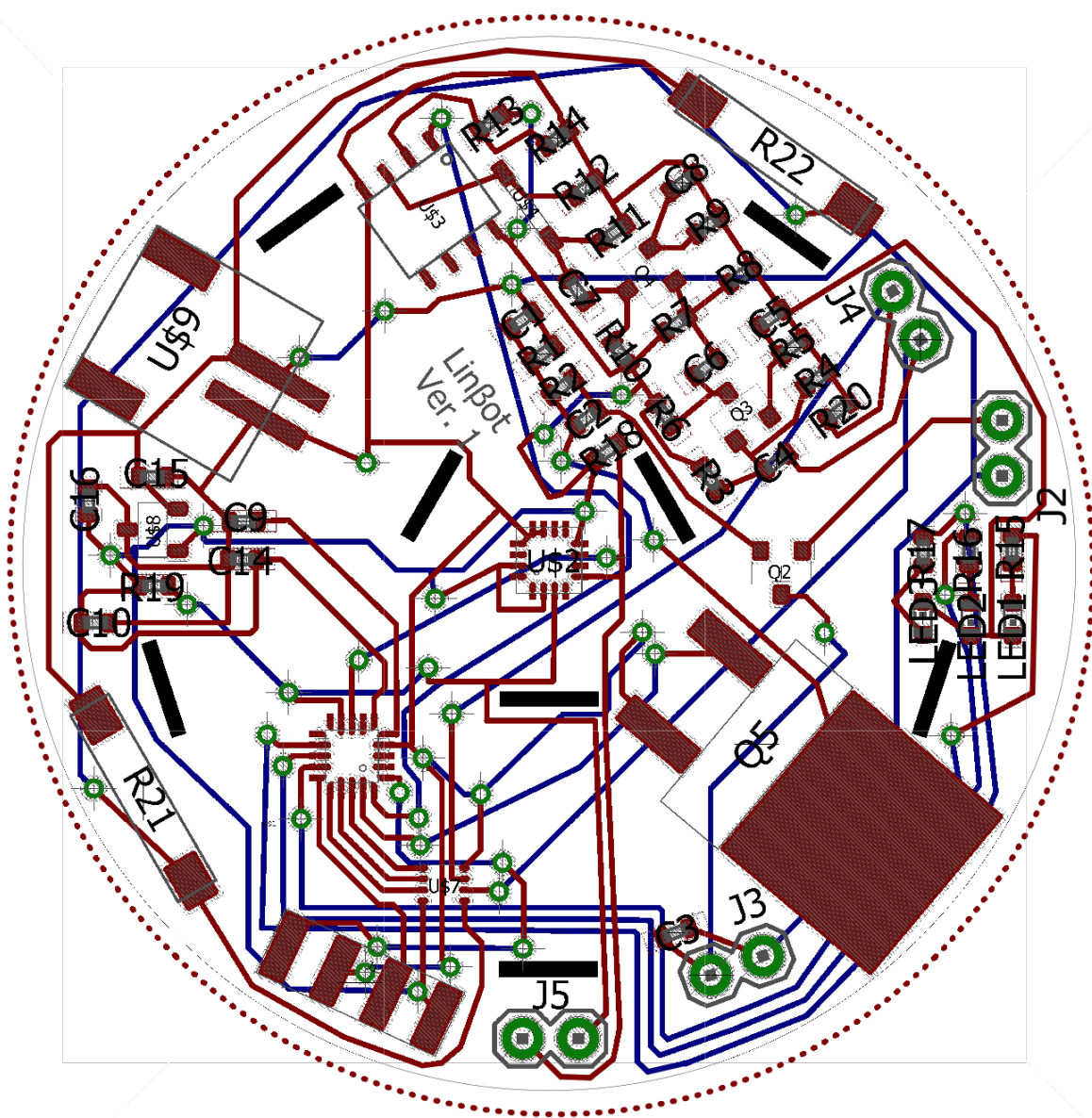


Figure C.1: Linbot board. Component list in Figure B.1

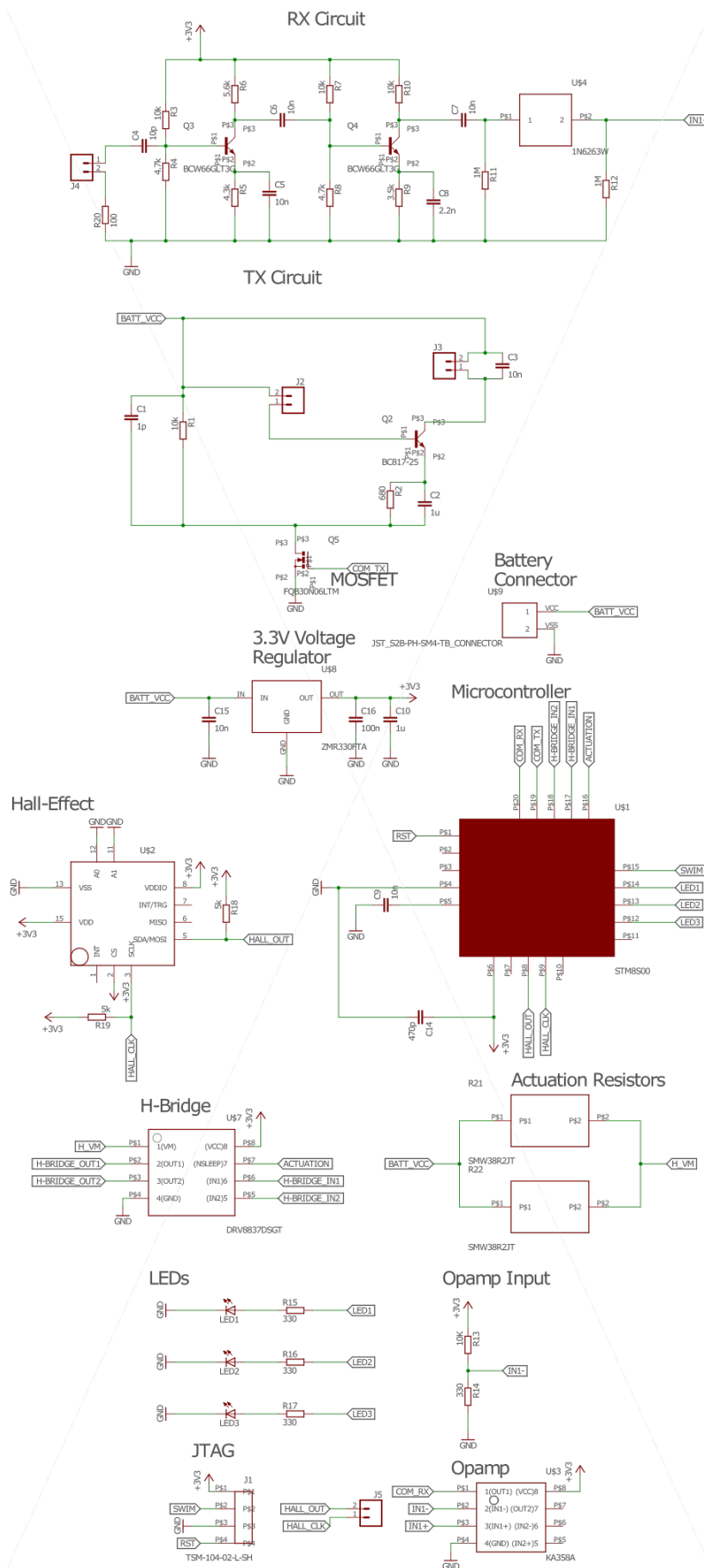


Figure C.2: Linbot schematic

Appendix D

Designing the Linbots

D.1 Mechanical Design

I started with the design of a Wormbot actuator [33] and identified what needed to be changed:

- The soft, silicone spring used required a complex and time-consuming 3D soft lithography process to create.
- The distance between magnet and coil reduced the maximum force that could be produced.
- The two end caps, which functioned as the magnet and coil holders, had to be 3D printed. They also needed to have their support material removed manually. These factors lead to a large amount of time being spent making each one.

I experimented with different designs and materials to improve the Wormbot design. In order to make the fabrication process as quick and cheap as possible 2D processes like laser cutting were focused on. I started experimenting with paper and foam for the spring as these materials are soft, readily available and can be quickly cut. Foam cut into rings and stacked to create a spring proved to have a very small range of motion for its length. The paper required folding into a structure in order to provide a spring force. I tried cutting out individual paper strips and placing them unfolded around the sides of two end caps. This provided a small restoring force but it was difficult to make the legs buckle outwards appropriately. The next step was to place a fold in each leg to bias the direction of buckling. This worked but the process of attaching each leg was time-consuming. A design similar to the final kirigami design, shown by Appendix A, where the legs are all connected by a strip at the top and bottom was created. This allowed for a single structure to be created by rolling the cut out spring and easy attachment by glueing the end caps inside the two strips. This design showed some promise but ended up requiring multiple layers wrapped around each other to produce a reasonable restoring force. I decided to try acetate in place of paper as it is readily available and stiffer than paper while still being easy to fold. I selected acetate spring as for the final design.

My focus then moved to the end caps and attaching the spring to them. The process currently involved having the end caps fit at either end of the spring and glueing them in place. This meant that if the spring had been wrapped incorrectly and glued then

the end caps would not fit and a new spring would have to be made. In order to make sure the spring was the same diameter every time I introduced a barbed tab and a slot for it to fit into. This was quicker than glueing and allowed the spring to be uncurled without damage. This encouraged us to try using these barbed tabs instead of glue for more of the structure. This would mean introducing slots to the end caps.

The end caps needed to be redesigned completely so that I would not need to use 3D printing. Initially, I tried to design end caps that had integrated magnet and coil holders but no kirigami design seemed to suit the shape required. I created designs similar to the final designs, shown by Appendix A, but that had 6 slots on their base. This allowed 3 tabs that were longer than the diameter of the end of the spring to each pass through two opposite slots on the base of the holders and reconnect with the other side of the spring. This formed a lattice that supported the holders without the need for separate end stops. While this meant fewer parts it proved too flexible and most of the movement of the magnet and coil just bent the supports rather than contracting or expanding the spring. Rigid end caps to hold the components would avoid this problem. I also had to consider where the electronics were going to be placed, as the end caps needed to be rigid the PCB could be used as part of the structure. The other end cap was designed as a circle with slots for structural tabs to be cut out of rigid plastic.

D.2 Integrated Magnetic Communication

I searched for a suitable design for my coil communication system. Resonant circuits can be used to amplify alternating current through a coil which made them ideal for use in my communication system. Resonant circuits are made of a coil and a capacitor connected at both ends. When this circuit is driven at its resonant frequency it produces much larger current oscillations [50]. This decision also suggested that an LC oscillator should be used as it incorporates a resonant circuit and always oscillates at the circuits resonant frequency [58]. The first step in building this oscillator was to find the inductance of the coils I would be using as it can be used to choose a resonant frequency for a circuit using the equation:

$$f_{res} = \frac{1}{2\pi\sqrt{LC}} \quad (\text{D.1})$$

Where f_{res} is the resonant frequency, L is the inductance and C is the capacitance of the capacitor in the resonant circuit. I used a known C and drove the circuit via a signal generator. I then moved to simulating an LC oscillator as a number of components needed to be rapidly changed in order to find the correct values to create oscillation. During this time I realised that it would be better to use a smaller coil in the resonant circuit and then magnetically couple the oscillations into the main coil. These two coils would function as a transformer in order to produce higher voltage oscillations.

Once a simulated solution was found I built a version of the circuit on a breadboard so that I could easily change components. The real world performance was much worse than predicted, the LC oscillator completely failed to work. By changing some components in order to increase the amplification of the oscillations in the trigger coil I got the oscillator working. However, the oscillator produced a smaller amplitude wave than the simulated version. In order to better understand the circuit, I changed the simulated design from

just representing the coils as inductors to representing them as inductors and resistors. This brought the simulated effects in line with the real world ones. Using the simulated circuit I searched for ways to improve the resonance and settled on introducing a second transistor to create a “Darlington Pair” [59]. The pair of transistors would amplify the resonance. I introduced this change to the real circuit and there was less improvement than expected.

With the transmission now working I moved on to the receiving side of the communications. I decided to connect the main coil to the receiving circuit as, of the three coils, it should produce the largest voltage from magnetic oscillations. This is due to the induced voltage (V) being:

$$V = NA \frac{dB}{dt} \quad (\text{D.2})$$

Where N is the number of turns in the coil, A is the cross-sectional area of the coil, B is the magnetic field strength and t is time. Therefore, the coil with the most turns will produce the largest voltage oscillations when the magnetic field oscillates.

This means that the receiving and actuating circuit would be connected to the same coil but the h-bridge fully disconnects the coil from the actuation circuit whenever it was not in use, which means that receiving is not affected while not actuating. In order to turn the received carrier waves into output data, I would need an envelope detector [60]. This detector required that the signal flow through a rectifying diode and so would need to be larger than the diode’s forward voltage. I first decided to reduce the forward voltage by using a Zener Diode [61]. However, I still needed to amplify my signal as it was in the tens of millivolts. I chose to use two class A amplifiers in series to get the signal to a high enough voltage.

D.3 Sensing

In addition to tactile sensing, I decided to allow for extra I²C devices to be attached modularly via a pair of pins attached to the I²C bus. The tactile sensing is built in due to it also functioning for odometry and so being a key component of the system. The decision to have other sensors attached on through I²C pins is to minimise the cost of the Linbots and reduce the chance of them having unnecessary functionality for some tasks. The hall effect sensor was tested on a breadboard before being used in the PCB design.

D.4 PCB Design and Fabrication

I originally designed the PCB to be the same size and shape as the end cap, a 30cm circle. However, this proved too small to hold all of the components needed and so it’s and the end cap’s size was increased to 48cm. I first attempted to fabricate the PCBs in the lab. I successfully milled the PCBs and attempted to solder the components. when it became clear that due to the small size of the components I would not be able to solder them all by hand I investigated solder paste masking. This technique involves cutting a mask with gaps where there are component pads on the PCB, placing the mask over the PCB and spreading solder paste over it. Once the mask is removed

components can be placed onto their pads and will adhere due to the solder paste. The PCB is then placed in the oven and the solder reflowed. I decided to use vinyl to make the masks as it could be cut on an available laser cutter and the adhesive layer worked well for holding the masks in place. I experimented with the mask hole size relative to the pads. There was a trade-off as larger holes allowed more solder paste and improved the chances of the component soldering properly but also increased the chances of small nearby pads being connected by excess paste. Having found a good masking technique I needed to find a way to connect both sides of the board through vias (conductive holes in the board). I had access to a through hole plating kit but required a refill of the paste used to create the conductive plating. The cost of the paste was comparable to just having the boards made professionally, but not assembled, which would also come with the bonus of better quality boards. I ordered and assembled the boards for testing.

This first PCB design required changes due to an error in the TX system. This error was created when testing the Surface Mount Technology (SMT) transistors (with soldered on wires) with the existing breadboard circuit. I needed to test new transistors as the version I was using was not available with SMT. A wire was misplaced during this change leading to the first transistor in the Darlington pair being bypassed and the circuit being returned to its single transistor version. This mistake was not noticed as by coincidence the change in transistor model actually improved the function of the single transistor TX circuit so that it was now slightly better than the Darlington pair with the previous transistors. The change in transistor model also caused the Darlington pair circuit design to cease working. When the first PCB was built the TX failed to work and, having tried to solve the problem on the PCB, I examined the breadboard to work out why. The mistake was noticed and the existing boards were edited to bypass the problem.

My testing of the systems revealed that compressing the Linbots reduced their communication range. This is due to more of the magnets being positioned inside the communication coils. Filling the inside of a coil with some types of metal increases the magnetic field strength of electromagnets, and therefore would increase the range of the coils. Neodymium, however, wastes much of the energy of the magnetic field [62]. In order to overcome this, I redesigned the coils to have the communication components at the base and so be as far away from the magnetic field as possible. My change allowed nearest neighbours to communicate even when compressed.

I ordered boards of the new design and assembled them. Having made some changes to component values I decided that this design would meet requirements and ordered 10 more boards. When they arrived I assembled all 10 at once in a parallel process. Unfortunately, I did not use enough solder paste per board during this process and most came out with defects. This led to needing a much larger time investment fixing all of the boards individually. This was facilitated by a hot plate with a large metal disk to dampen its temperature oscillations. The boards could then be placed on this plate, their solder melted and the components could be manipulated. In total 9 of 12 boards have been fixed and I have moved forward using these 9 for my experiments.

Appendix E

Fabrication

Fabrication Steps

A full bill of materials with technical details can be found in Appendix B. The following components are needed for us to fabricate each board:

- A PCB . The PCB follows standard design rules so it can be fabricated by most PCB manufacturing companies.
- The electronic components.
- A section of acetate large enough for the designs.
- A section of acrylic of around 50x50mm.
- Wire for the coil.
- Permanent magnets (I used 3 stacked to create the correct shape).
- A 50x50mm section of vinyl.
- Solder paste.
- A spool of PLA for a 3D printer.

The following is my process to prepare the components for each robot:

1. I cut a solder paste mask from the vinyl.
2. I use the mask to apply solder paste to the PCB .
3. I place the electronic components onto the board
4. I reflow the solder in an oven (I designed these initial steps and the PCB so that this process can use pick and place technology and be fully automated if manufactured at a larger scale).
5. I cut out the spring and holders from the acetate either as single items or in a strip using a laser cutter.
6. I cut the end cap out of acrylic using the laser cutter.

7. I 3D print a coil holder mount (design in supplementary). The coil holder mount is currently the least manufacturable part although it can be made on a lathe or injection moulded. Only one coil holder mount is needed to produce any number of robots as it is reusable. The coil holder can also be mounted on any cylindrical object of appropriate size or held on a finger if hand wound.

From this stage, no specialist equipment is needed and while pliers and a hand drill make assembly faster the whole process can be completed by hand. my process to construct each robot is:

1. I wrap the coil holder into its 3D shape and push its side tab into its slot.
2. I mount the coil holder and attach its mount to my coil winding machine. The mount can also be attached to a hand drill or the coil can be wound by hand.
3. I wind the coil onto the holder.
4. I attach the coil holder to the bottom of the PCB by pushing the tabs through the central slots. Pushing tabs is easier with tweezers or pliers to allow for a better grip.
5. I curl the magnet holder into its 3D shape and push its side tab into its slot.
6. I insert my magnets and catch the fold over the tab (on top of the holder) between the top two magnets.
7. I attach the magnet holder to the cap by pushing the tabs through the central slots.
8. I attach the spring to the cap by pushing its tabs through the cap's radial slots.
9. I attach the PCB to the other side of the spring pushing the spring tabs through the PCB radial slots.

Appendix F

Supplemental Information of Publications

F.1 Linbots: Soft Modular Robots Utilizing Voice Coils Supplementary Data

Supplementary Data

The Electronic Design of the Linbot

Our communication system contains an inductor-capacitor oscillator, which sustains oscillations at the resonant frequency of the circuit and allows us to produce a carrier wave from a direct current source. Our oscillator uses two 12-turn coils; one coil is used in a resonant circuit to produce an oscillating magnetic field for communication and the other coil, a trigger coil, is coupled to this field and controls the gain of a bipolar junction transistor that drives the resonant circuit. We demonstrated successful communication between Linbots using on-off keying of a 700 kHz carrier signal and a baud rate of 1000 s^{-1} . The transmission circuit uses an MOSFET transistor (which is driven by the microcontroller) as a switch to turn the transmission on and off. The transmission signal is stepped up into a higher voltage by the larger 200-turn coil.

The oscillating field induces a signal in the millivolt range in the coils of nearby Linbots. We amplify the signal by passing it through two cascaded class-A amplifiers. We then pass the signal through an envelope detector and then feed it to a comparator. We set the comparator to use a threshold of 16 mV to create a square wave matching the transmitted data. The peaks of the noise generated by the receiving circuit can reach $\sim 14 \text{ mV}$, meaning that the comparator will only pull high on receiving a real incoming transmission, an example waveform of the noise is shown in Supplementary Figure S12. We pass this square wave signal to the microcontroller, where we can recreate the original message.

In addition to receiving the transmitted data, we use the 200-turn coil for actuation. Application of current to the coil allows the Linbot to be contracted or extended along its central axis from its rest position; the direction of actuation depends on the polarity of the current applied to the coil. The bidirectional actuation is shown in Figure 1D and E. The applied current induces a magnetic field in the coil, which either attracts or repels the permanent magnets resulting in this actuation mechanism. We use the microcontroller to control both the frequency of the actuation and the direction of the actuation via the H-bridge driver.

The Hall-effect sensor is controlled by the microcontroller through an I²C bus. There is an additional pair of header pins included on the Linbot printed circuit board (PCB). These header pins are connected to the microcontroller I²C peripheral and allow for extra sensors to be easily added to the Linbot.

We incorporated a programming port into the Linbot PCB. We program the Linbot via an ST-Link/V2 debugger using a single wire interface module interface.

The Fabrication of the Linbot

Cost and functionality were the most important factors considered when designing the Linbots. Our rationale behind the system design was to keep the costs as low as possible without sacrificing functionality. The cost of a single Linbot PCB is £7.70 in a batch of 10. The cost of the electronic components associated with a Linbot PCB is £5.94 in a batch of 10.

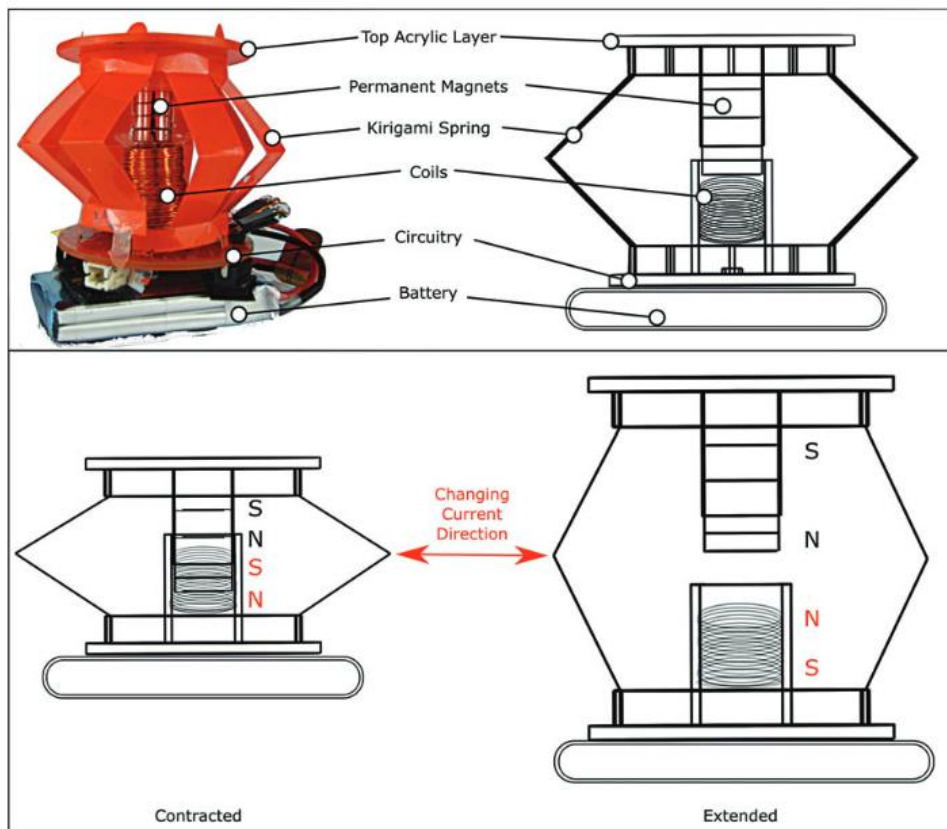
We purchased the 0.35 mm insulated copper wire and 10 mm permanent neodymium magnets from RS Components. Each Linbot requires three coils: one 200-turn coil and two 12-turn coils. We use our custom-made coil-winding machine, shown in Supplementary Figure S5, to wind these coils. The resulting structure has an inner diameter of 14.5 mm, an outer diameter of 18.5 mm, and a height of 22 mm. Our machine feeds the wire onto a rotating coil holder and we deposit superglue on the wire as it runs so that the coil holds its shape.

We fabricated the top layer of the Linbot from a 3 mm acrylic sheet. We cut the patterns for the kirigami components and top layer using a laser cutter (Epilog Laser Fusion 32). We designed the Linbot PCBs using Eagle PCB Design Software and fabricate them on double-sided Cu-FR4-Cu 0.1-mm boards using an external company called Ragworm (Kent, United Kingdom). We purchased the Hall-effect sensors, MLX90393 (Micropower Triaxis Magnetometer), from Mouser Electronics and soldered them onto the Linbot PCBs to provide sensing abilities.

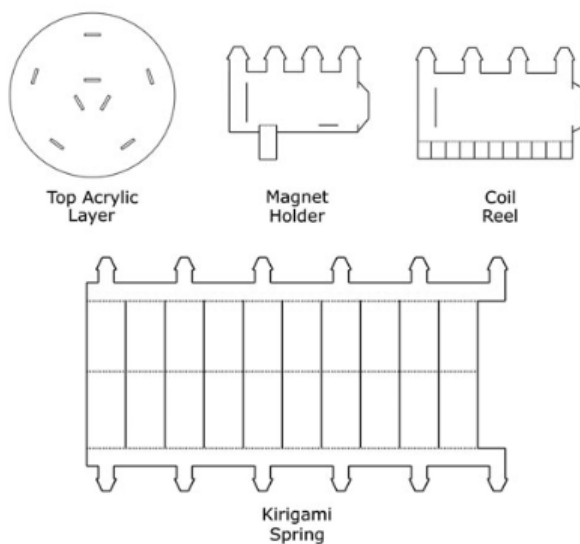
The Linbots have a minimum battery life of 22.5 min and a maximum of 280 h. The minimum battery life is calculated by assuming constant actuation, which uses a current of 1.2 A. The maximum battery life is calculated by assuming the Linbot is in sleep mode, where it consumes an average current of 1.6 mA. In this mode, the battery life of a Linbot is more than 11 days. The other Linbot capabilities have different current consumption levels and thus can change the battery life. Communication with other Linbots draws 80 mA. The current consumed by the Linbot when functioning as a speaker at maximum volume is 1.2 A. Using the Linbot for tactile sensing will use a current of 8.7 mA. Therefore, the battery life of a Linbot running any of these functionalities or running several functionalities together can be easily calculated. In this Linbot version, we detach the lithium polymer batteries for charging. We charge the batteries using a Linkman lithium battery charger and a 2S-6S balanced charging plate.

Technical Files

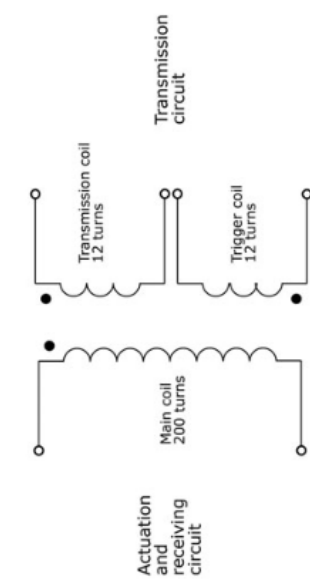
The authors have provided a zip-file for CAD files and PCB schematics.



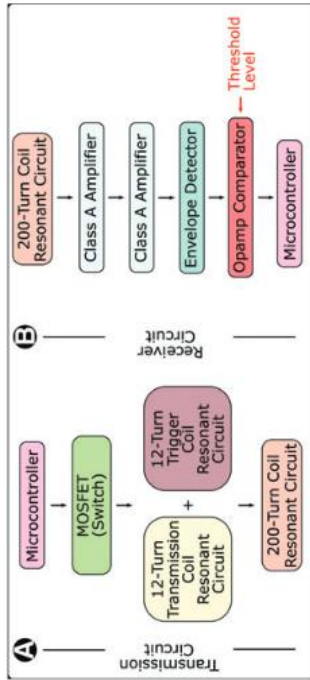
SUPPLEMENTARY FIG. S1. A labeled picture and sketch of the Linbot showing all of its components and a sketch of the actuation mechanism.



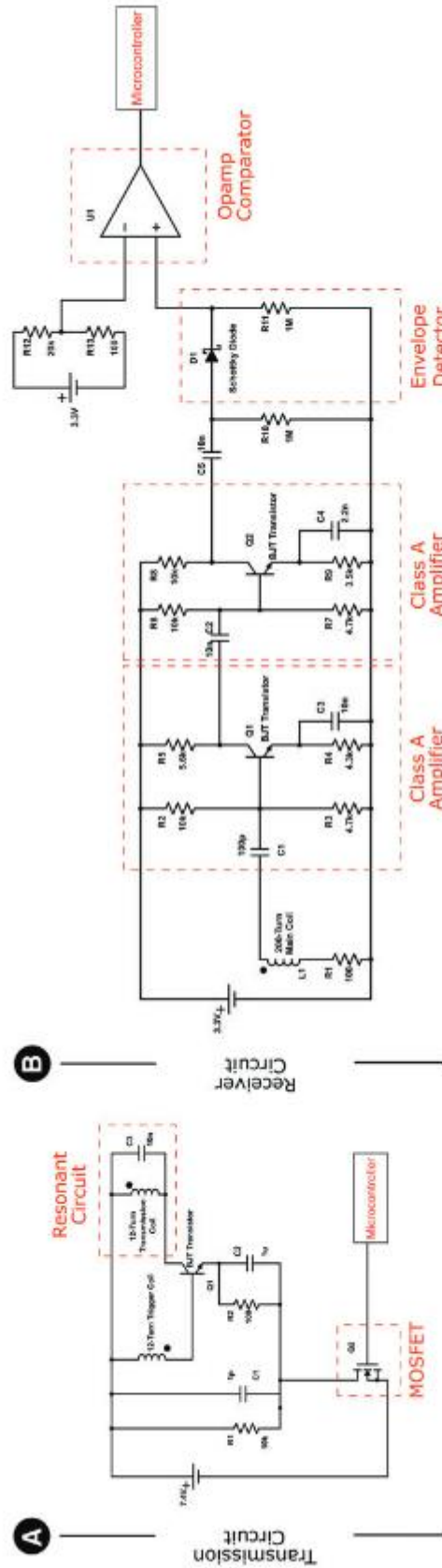
SUPPLEMENTARY FIG. S2. Two-dimensional design of the top acrylic layer, magnet holder, coil reel, and Kirigami spring.



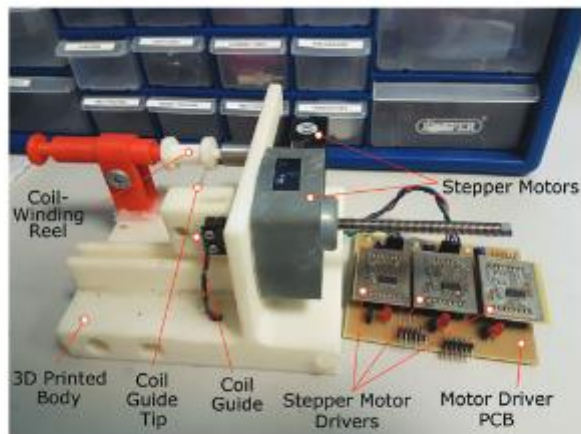
SUPPLEMENTARY FIG. S3. A circuit diagram of the transmission, actuation, and receiving coils.



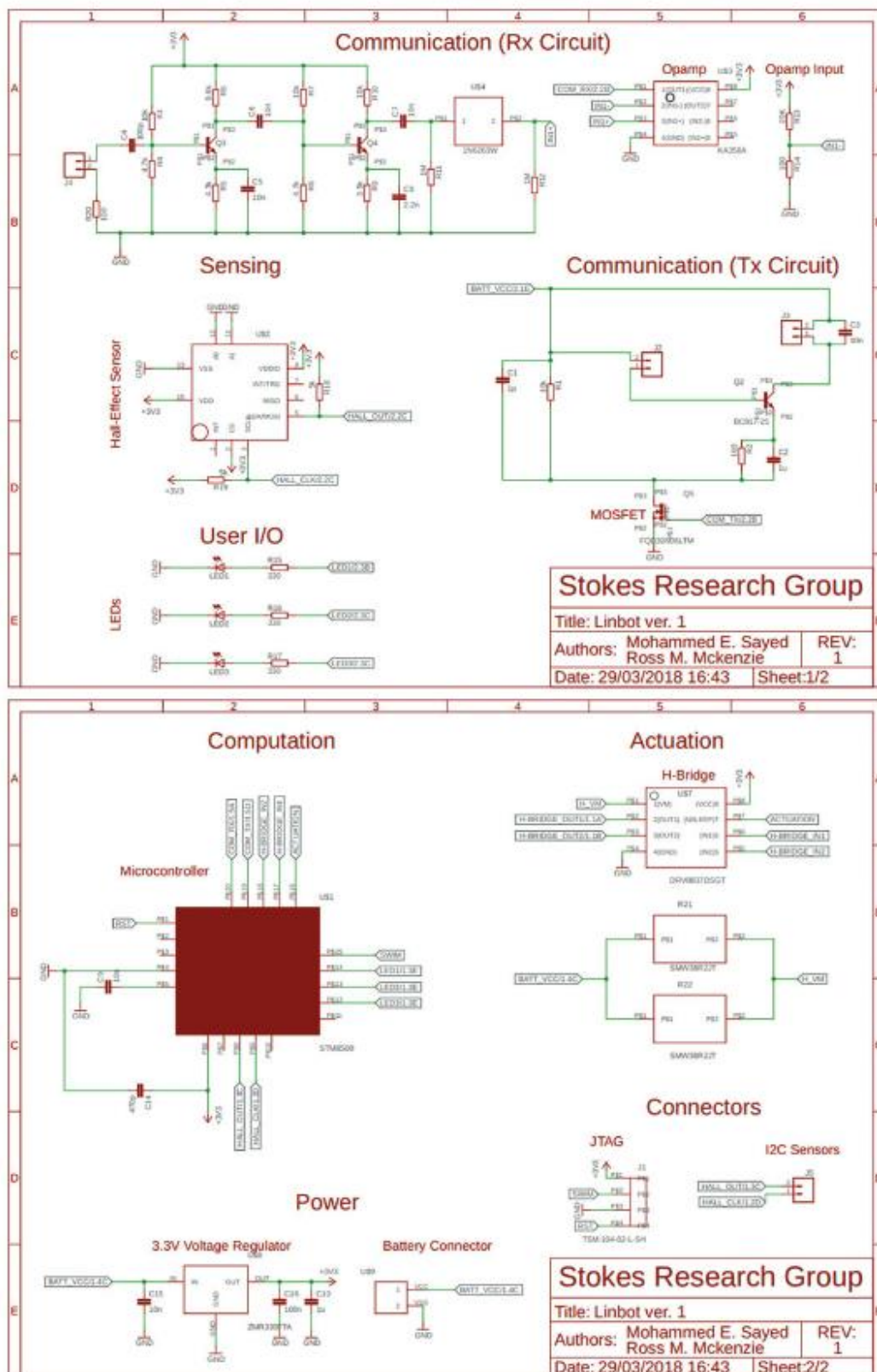
SUPPLEMENTARY FIG. S4. A block diagram of the (A) transmission and (B) receiver circuits.



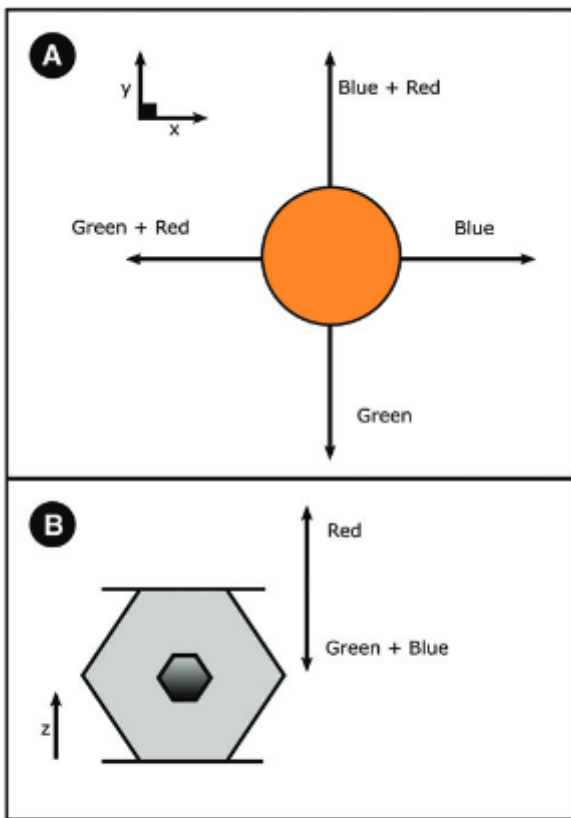
SUPPLEMENTARY FIG. S5. (A) Circuit schematic of the transmission circuit. The transmission circuit uses a MOSFET transistor (which is driven by the microcontroller) as a switch to turn the transmission on and off. The transmission signal is stepped up into a higher voltage by the larger 200-turn coil. The coupling between the coils can be found in Supplementary Figure S3. (B) Circuit schematic of the receiver circuit. The 200-turn coil is used for receiving the transmitted signal. We amplify the signal by passing it through two cascaded class-A amplifiers. We then pass the signal through an envelope detector and then feed it to a comparator. We set the comparator to use a threshold of 16 mV to create a square wave matching the transmitted data.



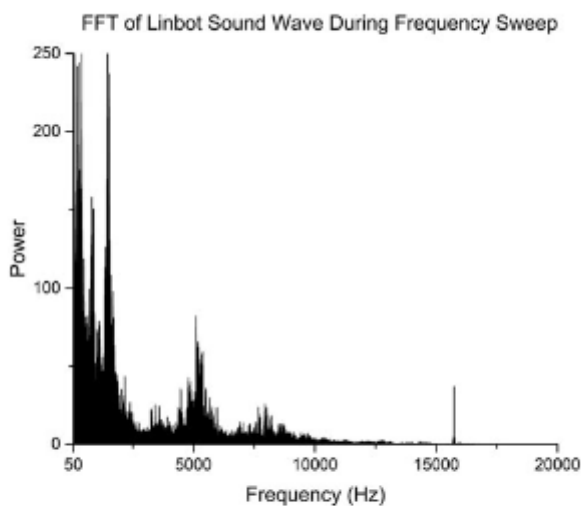
SUPPLEMENTARY FIG. S6. A labeled picture of the custom-built coil-winding machine used for producing the actuation coils of the Linbots.



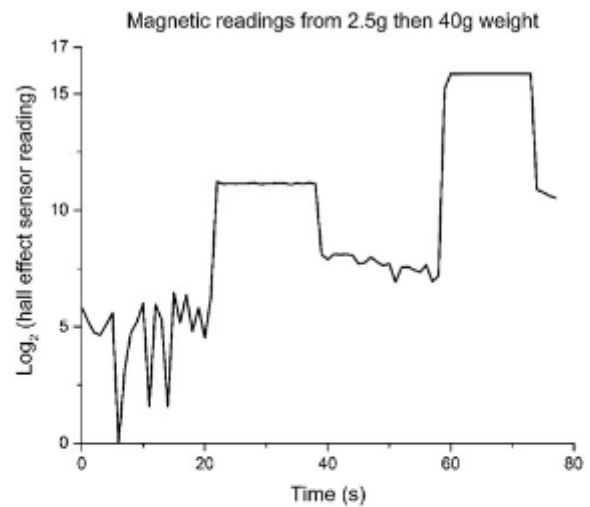
SUPPLEMENTARY FIG. S7. PCB schematic of a Linbot. PCB, printed circuit board.



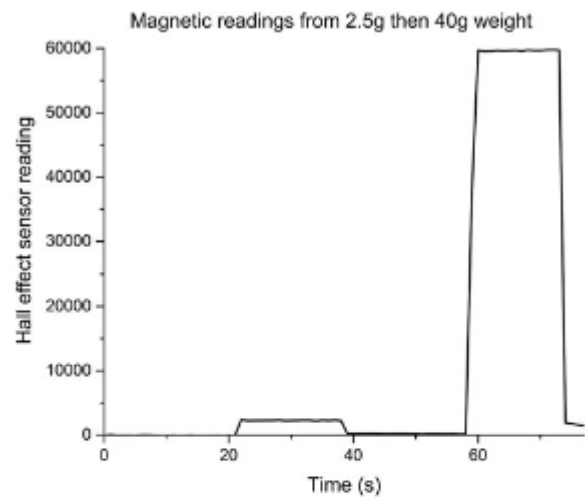
SUPPLEMENTARY FIG. S8. Tactile sensing experiment schematic showing the LED combinations displayed when the Linbot detects displacement (A) horizontally and (B) vertically.



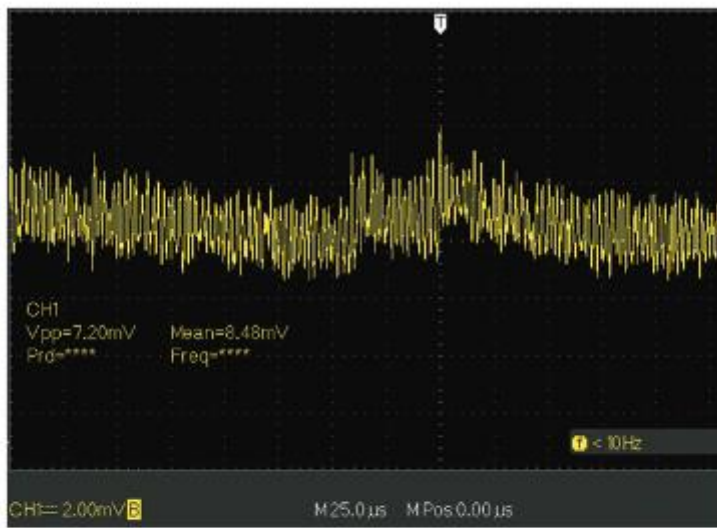
SUPPLEMENTARY FIG. S9. FFT of the sound wave produced by the Linbot during the frequency response experiment. The experiment is shown in Supplementary Video S3. FFT, fast Fourier transform.



SUPPLEMENTARY FIG. S10. A base 2 logarithmic plot of Hall-effect signal. The signal between 22 and 38 s represents a 2.5 g weight placed on a Linbot. The signal between 59 and 73 s represents a 40 g weight placed on the Linbot. The sampling rate was limited to 1 Hz by the connection between the Linbot and a computer, while the true sampling rate of the Hall-effect sensor is in the hundreds of hertz. The signal representing the Linbot with no weight on it changes after placing a weight on the Linbot.



SUPPLEMENTARY FIG. S11. A linear plot of Hall-effect signal. The signal between 22 and 38 s represents a 2.5 g weight placed on a Linbot. The signal between 59 and 73 s represents a 40 g weight placed on the Linbot. The sampling rate was limited to 1 Hz by the connection between the Linbot and a computer, while the true sampling rate of the Hall-effect sensor is in the hundreds of hertz. The signal representing the Linbot with no weight on it changes after placing a weight on the Linbot.



SUPPLEMENTARY FIG. S12. Background noise output from receiver circuit.

Appendix G

PeriSim README

perisim

Description

A peristaltic table simulation.

This package simulates a square grid of peristaltic cells beneath a flexible surface. Each cell is modelled as a gaussian disturbance in the flexible surface. Each cell can actuate to increase or decrease its amplitude. Objects on the surface then roll down the gradients of the surface.

The simulation can randomly vary its parameters in order to allow for controller optimization using the radical envelope-of-noise hypothesis [Evolutionary Robotics and the Radical Envelope-of-Noise Hypothesis, Nick Jakobi, 1997].

Any units can be used as long as the parameters are updated to be consistent, the default units are millimeters, seconds and kilograms.

Installation

```
pip install perisim
```

Usage

Make a perisim object with:

```
sim = PeriSim(x, y, cargo_pos)
```

The required arguments are:

`x`: The number of peristaltic cell columns of the table

`y`: The number of peristaltic cell rows of the table

`cargo_pos`: A list of initial XY coordinates of the cargo objects moving on the table
e.g `[[1, 2], [2,3]]`.

The parameters of the simulation are Keyword arguments:

`amplitude`: The height which a cell can expand or contract by. Default 5.

`spacing`: The spacing of rows and columns. Default 80.

`stddev`: The standard deviation of the Gaussian disturbance of each cell. Default 40.

`time_step`: The time step of the simulation. Default 0.01.

`variance`: The maximum proportion that a parameter can be randomly varied by at startup. Default 0.

`cargoVel`: A list of initial XY coordinates of the cargo objects moving on the table. Needs to be the same length as `cargo_pos`. Assigns 0, 0 for all if None. Default None.

`height`: The rest height of the gaussian disturbance caused by a cell. Allows for surfaces that have slight deformations while the table is at rest. Default 0.

`cargo_mass`: A list of the masses of the cargo objects moving on the table. Needs to be the same length as `cargo_pos`. Assigns 0.01 for all if None. Default None.

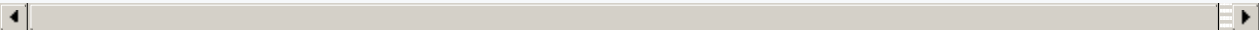
`g`: Gravitational strength. Default 9800.

`friction`: The frictional force as a proportion of current velocity. Default 0.01.

`act_force`: The proportional increase in reaction force experienced when on an expanding cell. Default 100.

`act_time`: the time taken for a cell to actuate. Only effects actuation force, not gradients. Default 0.1.

`gpu`: Whether the simulation should use gpu acceleration. Only noticeable on very large tables. Default False.



To run the simulation for one time step:

```
sim.update()
```

To change the actuation of a cell with grid coordinates (x,y):

```
sim.actuate(x, y, direction)
```

where direction is:

```
0 for rest height  
1 for extension  
-1 for contraction
```

To create a 3D visualisation of the system:

```
sim.visualise()
```

For an animated visualisation set the animate keyword to True and add an end time in seconds:

```
sim.visualise(animate=True, end_time=60)
```