UNIVERSITY OF CALIFORNIA SAN DIEGO

Synchronization through Manipulation - Indirectly Coupled Oscillating Walking Robots

A Thesis submitted in partial satisfaction of the requirements
for the degree Master of Science

in

Engineering Sciences (Mechanical Engineering)

by

Yudong Wei

Committee in charge:

Professor Nicholas G. Gravish, Chair
Professor Jorge Cortés
Professor John T. Hwang

2021

The Thesis of Yudong Wei is approved, and it is acceptable in quality and form for publication on microfilm and electronically.

University of California San Diego

2021

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I would first like to sincerely thank my supervisor, Prof. Nicholas Gravish, whose expertise was invaluable in guiding me to do some research work in the lab. Even at this difficult time, his continuous support and effort created a good research environment for us.

Then I would like to acknowledge the rest of my thesis committee members: Prof. Jorge Cortés and Prof. John Hwang for attending my thesis defense and teaching some of my most valuable and vivid courses at UCSD.

I also want to thank Zhuonan, Wei and all the members in Gravish Lab for inspiring me with creative ideas and helping me with my experiments. I really treasure those joyous days in the lab with them.

Last but not the least, I would like to thank my parents for supporting me finish my master study in UCSD and also want to appreciate my friends for the pleasure of their company in this wonderful academic journey.

ABSTRACT OF THE THESIS

Synchronization through Manipulation - Indirectly Coupled Oscillating Walking Robots

by

Yudong Wei

Master of Science in Engineering Sciences (Mechanical Engineering)

University of California San Diego, 2021

Professor Nicholas G. Gravish, Chair

Synchronization behavior universally occurs in our daily life, and it is so important and ubiquitous that has been studied over the past few decades. For example, the Millennium Bridge nicknamed as "Wobbly Bridge" because of the alarming swaying motion caused by synchronization of pedestrians on its opening day. Nowadays, along with the improvement of theoretical research, scientists start to control swarm of robots based on the idea of synchronization. However, no previous study has investigated the synchronization among indirectly coupled robots, which are completely independent with each other without any central controller. This thesis describes the design and implementation of a new kind of swarm of robots and their decentralized controllers. Firstly, robots with one rear driven legs are designed to form swarms of robots.

Then, an adaptive control method based on Hopf oscillator is implemented to regulate the phase of the oscillating walking robots. Data from both simulations and experiments has been collected and analysed to discover the properties of collective behavior. Finally, the significance of synchronization of multi-robot working systems is verified to be lower power consumption and greater total output force compared with disordered or non-regulated systems.

Chapter 1

Current state of art and Introduction

1.1 Current state of art

Synchronization is a ubiquitous phenomenon in the nature world. In spite the nature tendency for being disordered, synchronization is such a unique and abnormal phenomenon. However this kind of collective behavior is so universal that it occurs from non-living things (e.g. chemical reactions and turbulence) [1], to living things (e.g. fireflies [2, 3] and bacterial flagella [4, 5, 6]). With a large amount of research results from physicists, chemists, mathematicians and biologists, engineers are able to implement more engineering methods and improve our lives. For instance, the study of mutual synchronization occurs in our bodies including the pacemaker cells of the heart [7, 8, 9], the insulin-secreting cells of the pancreas [10, 11], and the circadian pacemaker [12, 13]give us a better understanding of our body mechanisms [14].

Animals and people can get certain benefits from synchronization. For instance, when trying to carry a piece of food which is dozens of times bigger and heavier than a single ant, the group of ants would gather into a formation and synchronize their motions to make impossible becomes possible [15, 16] (also as shown in Fig. 1.1)a. Another case is shown in Fig. 1.1b, a group of athletes paddle hard in a dragon boat competition, which is a traditional custom in China.

Only with perfect synchronization on the act of paddling [17], the boat is able to move fast enough to win the first place. Besides, Fig. 1.1)c shows a popular tradition in Alaska - the dog sledding. People can sit on a sled which is being pulled by dogs and speeding through the Alaskan forest. In this traditional event, it can be observed that all the dogs are almost synchronized in pace with each other to generate huge drag force simultaneously. It is known that the synchronization could greatly help the animals and people to work better, but the mechanisms behind is still unclear. Thus, it is of great importance to do more in-depth study on mutual synchronization, especially the collective behavior of living things.

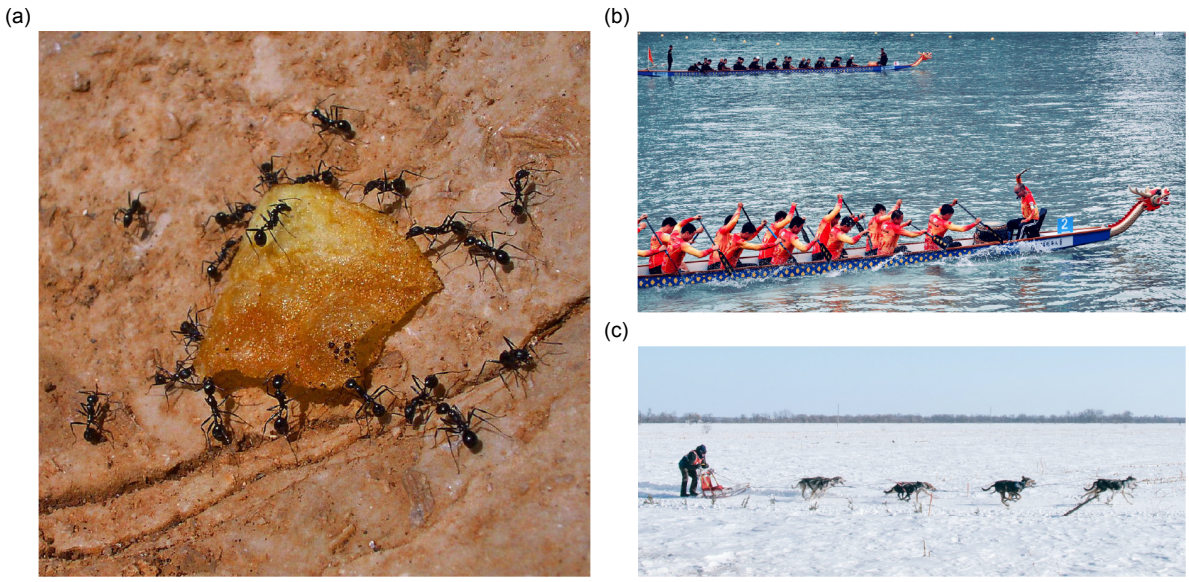

Figure 1.1: Some synchronization behaviors in the natural world and our daily life.

1.1.1 Previous Mathematical Approaches

To study the synchronization behavior in the nature world theoretically, we must have some mathematical interpretations and further analysis. Obviously, we can regard the periodic spacial locomotion as oscillators, which can be nonlinear and complex. In order to get rid of the difficulties by directly analysing a group of such oscillators, some further simplifications are

needed. Therefore, a great deal of previous research has focused on modeling and analysing the oscillators in the aspect of mathematics. Generally, there are two types of oscillators: smooth oscillators whose locomotion is continuous, and episodic oscillators which are pulse-like.

The most common representation of smooth oscillators is the limit cycle behavior [18]. There are many well-known limit cycle models like the Kuramoto oscillator [1, 19], the Hopf oscillator [20, 21], the Van der Pol oscillator [22], etc (detailed equations and properties are shown in Table 1.1).

Table 1.1: A table of three typical models describing smooth oscillators.

Name	Equation	Property
Kuramoto oscillator	$\dot{\varphi}_i = \omega_i + \sum_{j=1}^{N} \Gamma_{ij}(\theta_j - \theta_i)$	simple and abstract; only regulate the phase
Hopf oscillator	$\dot{\varphi}_i = \omega_i$ $\dot{r} = (\mu - r^2)r$	adjust both on phase and radius; perform perfect limit cycle behavior
Van der Pol oscillator	$\dot{x} = y$ $\dot{y} = \mu(1-x^2)y - x$	sharp corners on phase space; bent isochrones

The Kuramoto oscillator is a typical type of phase oscillators, which is the simplest and commonest way to describe and analyze weakly coupled oscillators. The Hopf oscillator (Hopf bifurcation) is a kind of isochronous oscillators, and this type of oscillator is commonly used to describe the oscillatory behaviors in biology. The Van der Pol oscillator is a relaxation oscillator. This type of oscillator in general does not have closed form solutions, and is often related to physical mechanisms that keep accelerating and decelerating periodically. In terms of control and robotics, different types of oscillator equations could be implemented so that engineers are able to adjust the phase or the radius to meet their demands.

Another important type of oscillators is episodic. Such oscillators are of special importance for biological oscillators, which exhibit pulse-like behaviors and often synchronized via impulsive stimulus from each other. For example, a group of fireflies can get synchronized by responding to the flash of each other and adjusting their cadences accordingly.

One of the most popular model of such oscillator is the synchronization model of integrate-and-fire oscillator, which was introduced by Peskin in 1975 [9] and developed by Mirollo and Strogatz in 1990 [14]. The model is described by a return map, in which all the oscillator are moving smoothly from the lowest state to the highest state, once one oscillator reaches the highest state, it would immediately "fire" and return back to the lowest state. At the same time all other oscillators would be stimulated by the "fired" oscillator and immediately jump up for a certain state difference but not higher than the highest state. For almost all initial conditions, it has been proven that the system eventually becomes synchronized.

1.1.2 Bio-inspired Robotics and Swarm Robotics

The idea of bio-inspired robot is to learn from nature and apply the optimized mechanisms to the robot design. This is an effective way to solve engineering problems and has brought about the widespread attention in robotic research in the recent years.

Firstly, a lot of research works have been carried out in the area of synchronization on bio-inspired robots. A well-known example of coupled synchronized robotics is the central pattern generators (CPGs) [23] that controls the locomotion in animals. For both invertebrate and vertebrate animals that can produce rhythmic patterns of neural activities, CPGs are neural circuits that work like the spinal cord and are able to generate coordinated patterns of periodic locomotion without any external inputs from higher control centers like brain. This bio-inspired concept can serve as a lower level controller in bio-inspired robots, so that smooth switches between different gaits like swimming and walking is enabled [24].

Another important area of bio-inspired robotics is the collective behavior of swarm animals. Although the collective behavior of swarm animals has been widely studied, e.g. ants gathered into a group to carry heavy foods or form like a raft to survive on rivers, swarm robotics is a fairly new field [25]. For instance, inspired by the communication and locomotion of living cells, a kind of particle robot has been designed to have decentralized controllers and coordinate

to migrate in swarms [26]. The carrying ability of swarm robots with force interactions is also important. With the knowledge from study and test of the resultant force from different kinds of robots in group, researchers have designed a team of "μTug" robots can pull 2000 times its weight on glass [27]. Another example is the "Kilobot", a tiny robot that moves with two vibration motors. With thousands of Kilobots, researchers can design and let the Kilobot swarm to rearrange to a typical shape or transport an obstacle fully autonomously [28, 29].

1.2 Introduction

In this thesis we mostly focus on the smooth oscillators, especially the Hopf oscillator. With the inspiration from biology, we can apply control methods on robots to imitate the activities and behaviors of creatures. Then by constructing and developing robot systems and analysing experiment results, we can not only make progress in the sense of robotics, but also in turn get a better understand of the mechanisms behind the biology behaviors. Therefore, our research work could play the role as a two-way bridge between biology and robotics.

There are already lots of research work about the collective robot behaviors. However, instead of studying directly coupled agents via sensory feedback, we designed robot swarms which are indirectly coupled and no need of a centralized controller. This kind of robot does not require any sensors except for the built-in sensor of its actuator, and it is "proprioceptive" of external force stimulates so that synchronization could be realized by collective force interactions among a group these robots.

The benefit from synchronization is the motivation of our work. Some preliminary calculation has been implemented as shown in Fig. 1.2. We have modeled the force of individuals to push a car and derived the max resultant force at different synchronization status. When pushers are perfectly synchronized, the maximum resultant force is almost three times larger than the asynchronous case. Therefore, it would be extremely useful if we can build pushing robots

that can synchronized with each other.

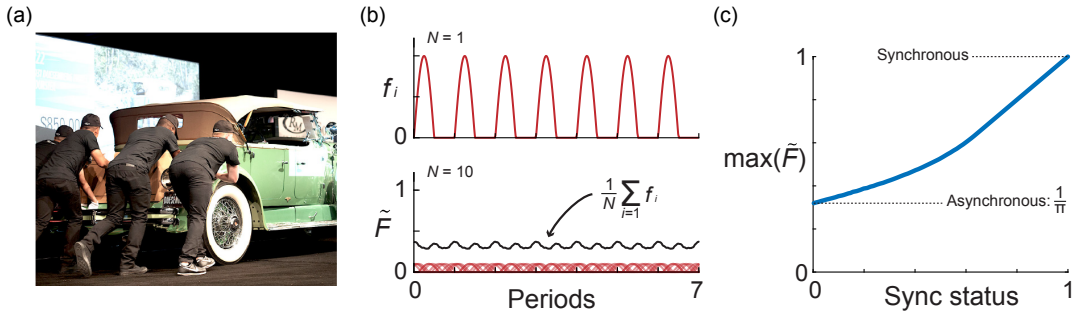

Figure 1.2: Collective pushing is optimized with synchronization of pushing forces. a) An example of people collectively pushing a car. b) We envision that individuals within the group apply a periodic force to push. The top plot shows the pushing force from a truncated sin wave in which the negative component is set to zero (i.e. there is no pulling force). Bottom plot shows the net force from a group of $N = 10$ pushers with random phase distributed between $[0, 2\pi]$. Force in both plots is normalized by number of pushers. c) The expected value of the maximum pushing force from a group is dependent on synchronization. Sync status is the amplitude of the complex order parameter defined in text. When pushers are in phase synchronized the maximum force is three-fold larger than when non-synchronized.

In Chapter 2 we describe the design work of the robot, including the mechanical structure, electronic components and the controller. The design of the platforms and the procedures of simulations and experiments are also described respectively in detail.

In Chapter 3 we demonstrate and compare the results from simulations and experiments. Some mathematical methods has been introduced to cope with the results. And some basic analyses are presented in this chapter. By comparison of results from simulations and experiments, we verify the simulation results with the experiment results.

In Chapter 4 we make some qualitative analysis from our previous results. The detailed mechanisms behind such indirectly coupled swarms of robots are studied to better understand the collective behavior and further improve our design of the adaptive controller.

In Chapter 5 we draw some conclusions as a summary of our current research work. Besides, we also make expectations on some improvement to be done in the future work and some further studies to be continued.

Chapter 2

methodology

2.1 Robot Design

To accomplish the goal of discovering the theories behind indirectly coupled robots, it is necessary to design the robots to be completely independent, easily manipulated and relatively small. Thus, we designed a tiny 2-legged robot (Fig. 2.1) with all components on it, so that this robot is not only independently controlled by its own controller but also fully isolated and autonomous among the swarm.

In general, this robot is mainly composed of two parts: lower mechanical structure and upper electronic components. Each robot's geometric dimension is about $65\,\text{mm} \times 30\,\text{mm} \times 35\,\text{mm}$ $(L \times W \times H)$ without electronic components ($65\,\text{mm} \times 30\,\text{mm} \times 60\,\text{mm}$ in total with electronic components), and its mass is 85 g in total.

2.1.1 Mechanical structure

The lower part is the actuator which contains all mechanical structure of the robot (Fig. 2.1b). To make the robot as simple as possible and easy to assemble, it was designed to have only one driven leg, and minimize the number of pieces of the whole robot. In this sense, this

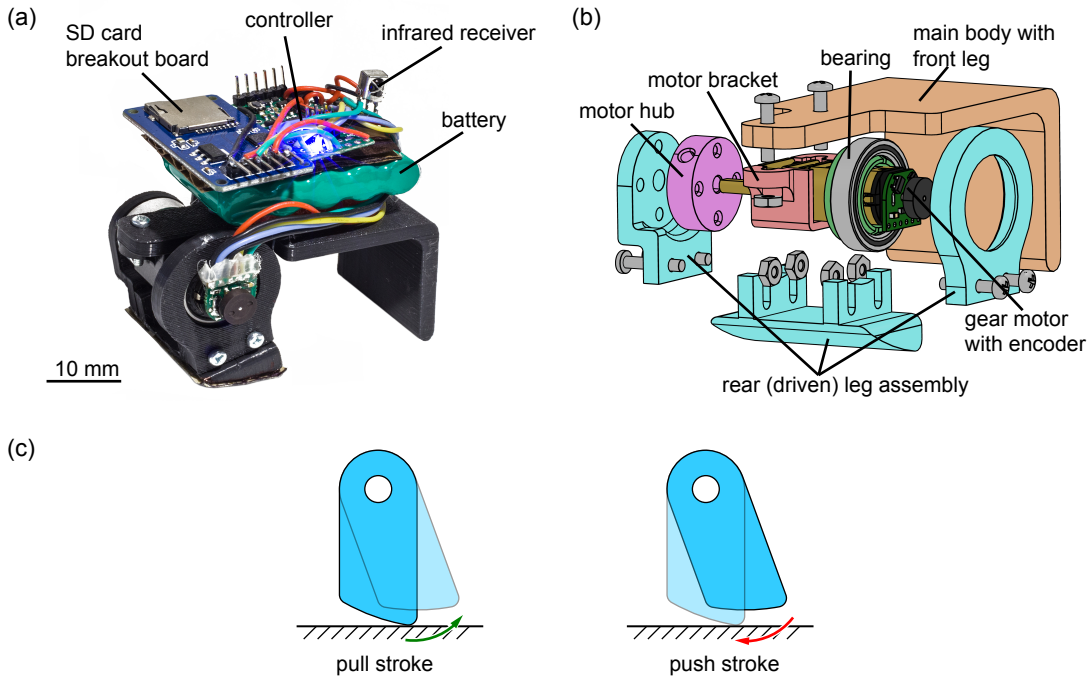

Figure 2.1: Design of the robot. (a) Complete structure of the robot with all electronic components. (b) Exploded view of the robot without upper electronic components. (c) Push-pull schematic diagram of the driven leg.

main mechanical structure of the robot contains a 3D-printed PLA main body with the front leg, a 3D-printed PLA rear leg assembly, a gear motor with encoder and other standard mechanical components. We designed the L-shaped main body to be both the chassis of the robot and the front non-driven leg to support the robot.

With one side connected to the motor output shaft and the other side supported by a bearing, the rear driven leg is composed of three parts. With this design, we can quickly redesign the shape of the foot and replace to the robot, so the process of the foot shape optimization is easier. With the intuitive of increasing the effective length of the leg while pushing forward, the foot is designed to have a inclined curved-shape foot for it to walking efficiently forward by simply push and pull gaits.

As shown in Fig. 2.1c, for each pull stroke, the foot starts almost at the position where the length between the rotation axis is the longest, then it rotates forward and tends to leave

the contact surface, and the robot almost stays still; while for each push stroke, the foot rotates backward, so that it generates forward driving force by pushing toward the contact surface and producing enough normal force. Thus, with repeatedly push-pull actuation, the robot can move forward.

2.1.2 Electronic components

The standard of choosing electronic components is as small as possible, so that we can fit every components easily on the robot body.

As for the actuator of the robot, it should be a component that can converts electrical energy into mechanical energy. There are plenty of choices like servo, DC motor, AC motor, solenoid and even piezoelectric actuator. For enabling the response to external perturbations, we finally use the brushed DC motor (from Pololu) with a 15:1 gearbox and encoder, so that both enough torque output and good backdrivability are ensured, and it has a tiny geometry dimension of $10\,\text{mm} \times 12\,\text{mm} \times 26\,\text{mm}$ and is $9.5\,\text{g}$ in weight.

To make each robot completely independent, all other electronic components are placed on the main body of the robot, including a controller, a SD card breakout board, an infrared receiver and a battery (Fig. 2.1a).

An A-Star 328PB Micro (ATmega328PB AVR micro controller) is chosen to be the main controller for the robot, which is almost the most tiny arduino controller but still has enough GPIO pins and PWM pins. It has a $5\,\text{V}$ regulator that can be powered from $5\,\text{V}$ to $15\,\text{V}$, and it features a $16\,\text{MHz}$ resonator, which offers good computing power.

According to the limitation on power output ability of the micro controller to drive the motor, a DRV8838 brushed DC motor driver is chosen to be the motor controller. It could output up to $1.7\,\text{A}$ continuous current and is capable with the power supply from $0\,\text{V}$ to $11\,\text{V}$. With the $5\,\text{V}$ PWM signal from the micro controller to determine the voltage output and a logic signal to define the rotating direction, the DC motor is capable to complete further research requirements.

Since the robot is designed to be fully independent, but the micro controller can not offer any wireless communication method, an infrared receiver is added. This infrared receiver is placed in the front of the robot's top surface for receiving the remote starting signal, so that all the robots in a group would be perfectly aligned in terms of time.

To record the results from the encoder and the controller, we use a SD card breakout board linked to the controller and save the results (e.g. measured position from the encoder and the time index) to a micro SD card. Therefore we can study the exact leg motion of the robot instead of simply record the videos of the experiment.

All the electronic components are powered by a 6.0 V NiMH battery pack placed between all the circuit boards and the main body. With its capacity of 350 mA h, the robots can finish almost 2 hours of the experiment when the batteries are fully charged.

With the electronic components above, the basic workflow of the robot controller is: 1. Receiving angular position information from the encoder; 2. Processing the results and implementing the designed control method to get output signal; 3. Send a Pulse Width Modulation (PWM) signal to the motor controller, then the motor is actuated. By implementing such workflow, the robot can move autonomously as designed.

2.2 Robot Control

In order to discover the collective behavior, especially the synchronization behavior of swarm robots, the control method must have properties of generating oscillating locomotion continuously, being robust against perturbations and interacting with other robots indirectly without additional sensors. Thus, we developed an adaptive controller based on the Hopf

oscillator in phase-radius coordinate system:

$$\dot{\varphi} = \omega_0 - K_{\text{adp}} f(\bar{\theta}) \tag{2.1}$$

$$\dot{r} = K_{\text{r}}(\mu - r^2)r \tag{2.2}$$

$$y = K_{\text{p}}(r\cos\varphi - \bar{\theta}) \tag{2.3}$$

where $\bar{\theta}$ is the angular position from the encoder, the output y is the voltage supplied to the DC motor by pulse width modulation. With constant ω_0 and μ and without the adaptive term following K_{adp}, the solutions of equations (2.1) and (2.2) are moving counterclockwise and eventually attracted to an limit cycle with a radius of $\sqrt{\mu}$, which are shown in Fig. 2.2. In equation (2.3), the output signal is composed of the difference between the desired angle $r\cos\varphi$ and the measured angle $\bar{\theta}$ multiplied by a proportional gain K_{p}, so the output equation is basically a position feedback controller which simply use the proportional control method.

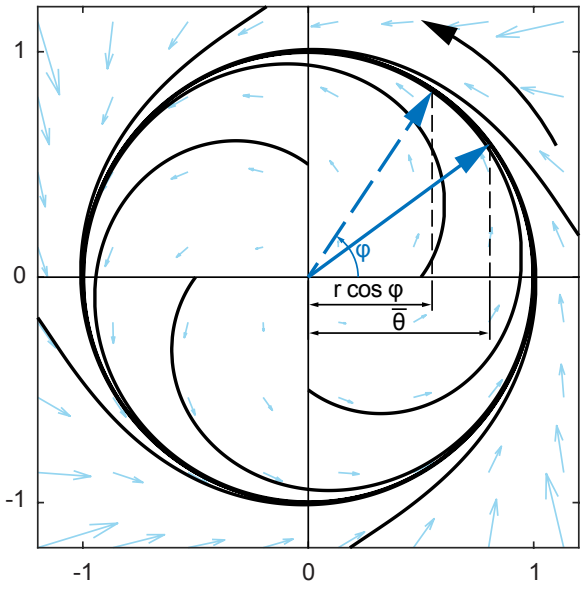

Figure 2.2: Phase-radius plot of the Hopf oscillator when $\mu = 1$, $\omega_0 = 1$, $K_{\text{adp}} = 0$ and $K_{\text{r}} = 1$ (the blue solid arrow line is the real phase correspond with the measured angle $\bar{\theta}$, while the dashed arrow line represents the target phase φ).

In practice, the robot should perform a smooth start and then generate constant amplitude of the phase , such that the gait would be stable for the robot to move forward. Thus, we set initial

condition $[\varphi, r] = [0, \sqrt{\mu}]$, so that r is constant, and the motion would start smoothly since the starting speed $\frac{d}{dt} r\cos\varphi = -r\sin\varphi\dot{\varphi} = 0$. With the properties of this controller, the robot's rear leg is able to perform back-and-forth oscillating motions with the stroke angle of $2\sqrt{\mu}$. Hence, the original adaptive Hopf controller has been simplified to an adaptive phase regulator.

In the aspect of "proprioceptive" feedback of the robot, the armature current of the DC motor can be calculated as:

$$I = (V - E_{\mathrm{b}})/R \tag{2.4}$$

Where V is the supply voltage, E_{b} is the back EMF and R is the resistance of armature. Since the rotational speed of the DC motor is relatively low in our usage scenario, and E_{b} is almost zero since it is proportional to the rotational speed, we can consider I to be proportional to V. Moreover, for DC motors, the output torque is directly proportional to the armature current I. Thus, we can regard the output torque τ as directly proportional to y, and define:

$$\tau = K_{\tau}/K_{\mathrm{p}}y = K_{\tau}(r\cos\varphi - \bar{\theta}) \tag{2.5}$$

With the idea of slowing the robots with ahead phases to manipulate them into phase synchronization, we attempt to choose $f(\bar{\theta})$ in equation (2.1) to be relative to the torque output, so $f(\bar{\theta})$ is chosen to be $\bar{\theta} - r\cos\varphi$. In this case, this adaptive function is just proportional to torque output, and the controller is highly influenced by force interactions. Moreover, by reason of the practical needs for the robot to stay still during the pulling gait, we can further modify the control equations to simply study the force interaction during only the pushing gait. Therefore, the modified controller is a phase regulator with conditions:

$$\dot{\varphi} = \begin{cases} \omega_0 - K_{\mathrm{adp}}(\bar{\theta} - \sqrt{\mu}\cos\varphi), & \text{for pushing gait: } \sin\varphi \geq 0 \\ \omega_1, & \text{for pulling gait: } \sin\varphi < 0 \end{cases} \tag{2.6}$$

where ω_1 can be different with (normally larger than) ω_0 for ensuring that the robot's leg can retrieve fast enough, so that during pulling gaits, the robot almost stays still.

By using such torque control and feedback method along with the backdrivability of the

motor, our robot provides the potential to interact with the environment. Thus, the next step is to test, verify and improve the preliminary design of our adaptive controller.

2.3 Simulation Details

We implemented and visualized the behavior of swarm robots via PyBullet [30], a physics simulation engine. In order to discover the synchronization behavior under an common and generic circumstances, we set up a certain amount of the 2-legged robots to push abreast against a large cubic obstacle (as shown in Fig. 2.3).

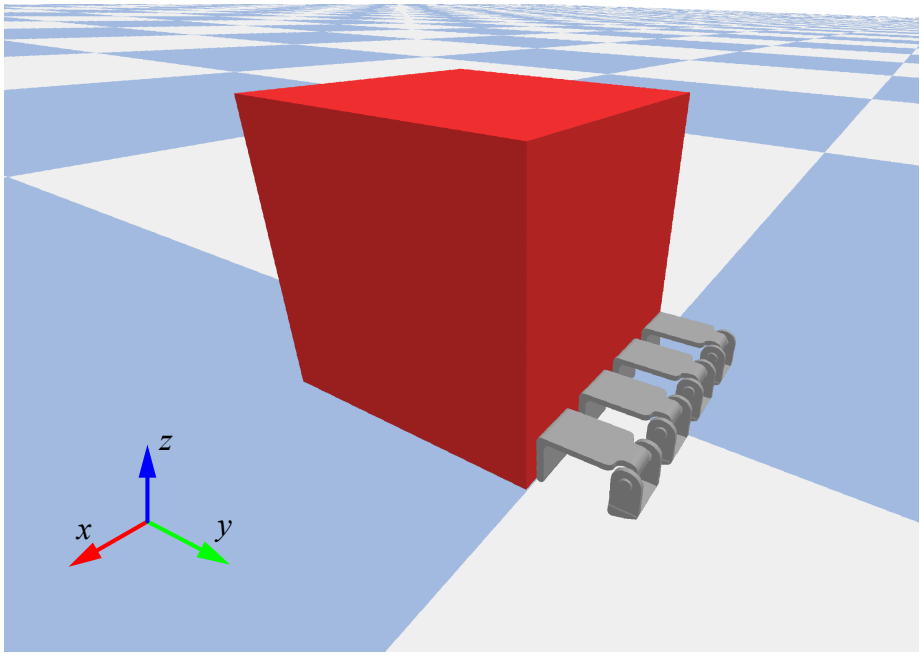

Figure 2.3: Simulation setup demonstration in PyBullet: four robots (as shown in grey) pushing against a cubic obstacle (as shown in red).

As for each robot, we firstly built a 3D model in Solidworks to have same physical properties as the real robot shown in Fig. 2.1a. The model has the geometric dimension of $65\,\text{mm} \times 30\,\text{mm} \times 35\,\text{mm}$ $(L \times W \times H)$, with its mass of 85 g in total, which are the same as the real model. Then, the model was exported as a Unified Robot Description Format (URDF) model (a collection of files describing robot models in ROS [31]) with its geometry and physical

information.

In Pybullet, all the URDF files were imported and ready for further control purposes. We set up the acceleartion of gravity as $10\,\mathrm{m/s^2}$ along the z-axis, and we set the coefficient of friction between the front/rear foot and the ground to be 0.7/0.4 as measured from the real robot. For the parameters in equations (2.6): $\omega_0 = 15\,\mathrm{rad/s}$, $\omega_1 = 15\,\mathrm{rad/s}$ and $\mu = 0.0625\,\mathrm{mm^2}$. We used torque control mode to simulate the motor output as equation (2.5), the torque gain $K_\tau = 0.1\,\mathrm{N{\cdot}m/rad}$. With the initial condition $[\varphi, r] = [0, \sqrt{\mu}]$, r stays constant at $\sqrt{\mu} = 0.25\,\mathrm{rad}$.

With a large cubic obstacle placed in front, the robots can push together side by side. The friction coefficient between the obstacle and the ground is set to be 0.5. Thus, we could then discover detailed synchronization information by varying the conditions like the initial condition of robots, the number of robots, and the friction between the obstacle and ground.

2.4 Experiment Details

In order to implement the controller in the real robot and verify the simulation results, an experiment platform is in need. Unlike the setup in simulation, in the real physical world, many factors become imperfect, e.g. the inconsistency of motor among different robots, the difference of friction for each robot's foot, etc. So that on the one hand, there are more detailed factors which should be taken into consideration, and it is necessary to establish some constrains to simplify the experiment. On the other hand, the experiment is also good to show how the robot would behave under imperfect circumstances and outside perturbations.

With the above considerations, we set up the experiment platform as Fig. 2.4. In addition to the basic setup in simulation, the robots is set against a "wall" attached to two linear shafts, then the pushing force from the robots could be transferred to the cuboid obstacle. In this way, the rotational motion of the obstacle can be constrained, then only the motion and force along the moving direction need to be taken into account. Furthermore, the motion of robots are also

constrained by several acrylic bars in parallel, these acrylic bars act as rails preventing robots from not walking along a straight horizontal line.

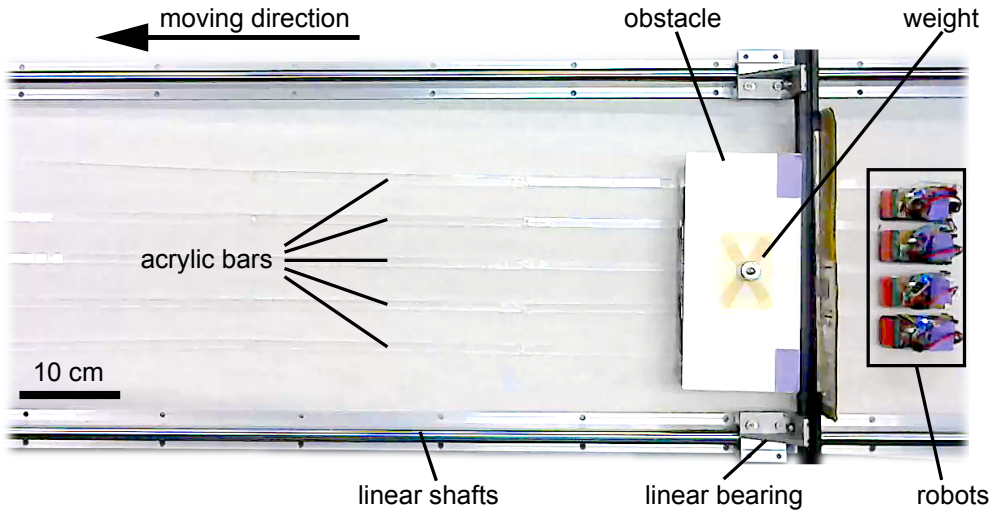

Figure 2.4: Experiment setup.

Since we use a geared motor with encoder as the actuator, some control parameters should be revised in comparison to the simulation. With some preliminary test on walking gaits, the parameters in equations (2.6) are: steady-state radius $\sqrt{\mu} = 12$ encoder counts, $\omega_0 = 15\,\mathrm{rad/s}$ and $\omega_1 = 30\,\mathrm{rad/s}$. Then the controller is ready to be implemented in the arduino for further experiment. Considering the computing power of arduino and the numerical accuracy, we simply implement the 2nd order Runge-Kutta (RK2) method to compute the controller equation. Before each experiment, the robot's leg would be firstly calibrated to an certain angle to its body as the equilibrium position, then rotate to $\sqrt{\mu}$ and wait for the starting signal.

Chapter 3

Results

With the controller prototype designed for the robots, the next aim is to implement the design and verify the functionality of the control method. In this chapter, some simulation results are firstly presented: including basic test on two robots; results from different numbers of robots, different adaptive parameters in the controller and different loads; comparisons of synchronization and disordered case in terms of energy consumption, total output force and working efficiency. Then, similar test results are obtained from experiments to verify the effectiveness of our simulation, and further experiments have been performed to also test the benefits we can gain from synchronization and seek for some real application scenarios.

3.1 Simulation Results

3.1.1 Basic tests on two robots

The first priority is to test the synchronization ability of out robot. The simplest and most intuitive way to do this is to test on just two robots. By manually setup a proper initial delay, these two robots could start with the specific phase difference that supposed to. Then, some basic results can be obtained from Pybullet simulations, e.g. two robots starts with almost an anti-phase

condition as shown in Fig. 3.1.

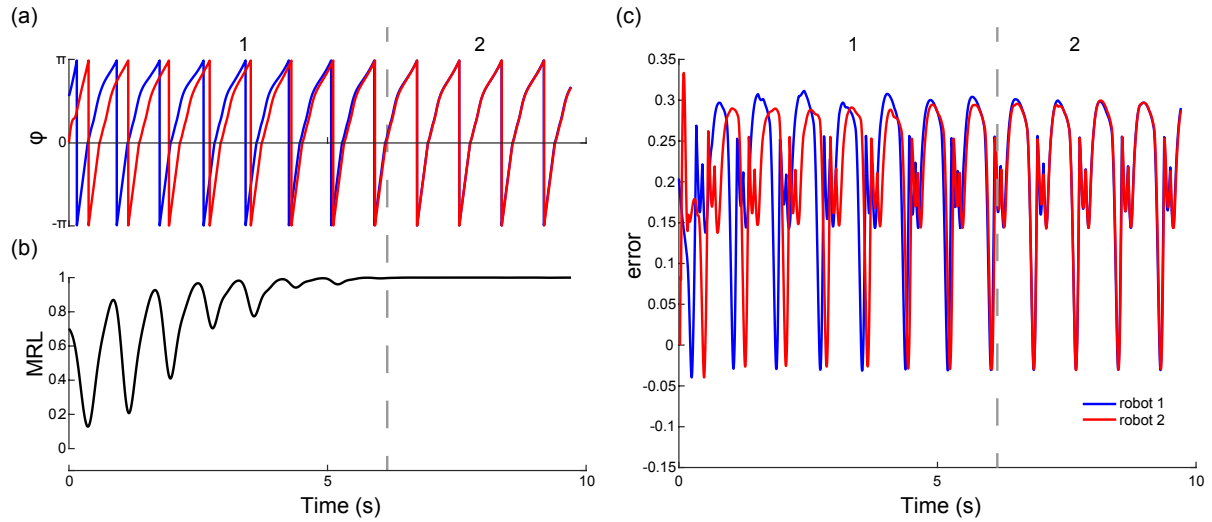

Figure 3.1: Some basic results from simulation on two robots. (a) Plot of phase φ for each robot (wrapped in radians to $[-\pi, \pi]$). (b) *Mean resultant length* (MRL) of two robots' phases. (c) The angular error $(\bar{\theta} - \sqrt{\mu}\cos\varphi)$.

All the results related to the robot dynamics can be sorted into mainly two types: internal phase/angle data from the controller and perturbation data from the outside. It is the combined effect of these two types of factors that leads to the collective behavior of two robots.

Internal data includes target angle ($\sqrt{\mu}\cos\varphi$) of the robot leg, phase φ from the controller, etc. To further analysis the result, phase data is mostly concerned since it is the most original and cleanest data and indicate the entire revolution of both locomotion of each robot and relationships among all robots, which is shown in Fig. 3.1a. It could be easily concluded from the plot that two robots stared at a different phase then gradually converge to a synchronized status. However, it is hard to judge how well the two robots get synchronized, i.e. to quantify the relationship between two robots so that we can intuitively study the synchronization revolution. This problem would become more serious in the case of a large number of robots. Thus, a circular statistic method called *mean resultant length* [32, 33] is been used to calculate the mean length of the resultant

vector. As shown in Fig. 3.2, the resultant vector can be calculated as:

$$C = \sum_{i=1}^{n} \cos\theta_i, \quad S = \sum_{i=1}^{n} \sin\theta_i, \quad R^2 = C^2 + S^2 \ \ (R \geq 0) \tag{3.1}$$

$$\cos\bar{\theta} = C/R, \quad \sin\bar{\theta} = S/R \tag{3.2}$$

in which θ_i is the angle of each vectors, the quantity of $\vec{R}$ is the length of the resultant vector, $\bar{\theta}$ is the direction of the resultant vector.

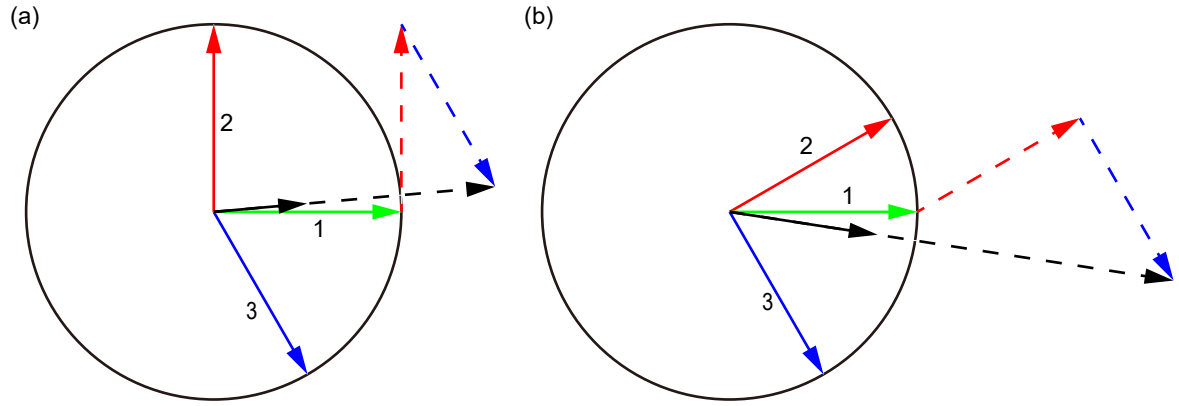

Figure 3.2: Example of *mean resultant length* in a unit circle: the length of one-third of the resultant vector (black dashed arrow), and the resultant vector of three unit vectors (in green, red and blue) is got by vector addition.

Then the *mean resultant length* can be simply defined as:

$$\bar{R} = R/n \tag{3.3}$$

which is the length of the resultant vector divided by the number of vectors. In Fig. 3.2a, $\theta_1 = 0$, $\theta_2 = \frac{\pi}{2}$, $\theta_3 = -\frac{\pi}{3}$, so that $\bar{R} = \frac{\sqrt{C^2+S^2}}{3} = \frac{\sqrt{4-\sqrt{3}}}{3} \approx 0.502$. Similarly in Fig. 3.2b, $\bar{R} \approx 0.798$.

With such statistic method, the degree of synchronization among robots can be represented by $\bar{R}$, which is a value in $[0,1]$. When all the robots' phases lies in the same group of vectors, we can consider that: $\bar{R} = 0$ means the phases are completely disordered or randomly distributed; while $\bar{R} = 1$ indicates all the phases are coincident and all the robots are fully synchronized in the same phase. And the closer $\bar{R}$ to one, the better the robots get synchronized with each other. As shown in Fig. 3.1b, the smoothed *Mean resultant length* data from Fig. 3.1a reveals the detailed process from out-sync to synchronization.

Another type of data is about the outside perturbation, which is the fundamental cause of synchronization behavior. The output force is the data related most directly to the outside perturbations.

With the analysis conducted in section 2.2, the angular error $(\bar{\theta} - \sqrt{\mu}\cos\varphi)$ from the proportional controller is directly proportional to the torque output of the motor. As for the force output to drive the robot forward, when the leg angle with the vertical line to the ground is relatively small and the distance between the motor axle and the contact point on the ground hardly changes, this driving force could be considered as directly proportional to the torque output of the motor. Moreover, the input term we designed in the controller is also $(\bar{\theta} - \sqrt{\mu}\cos\varphi)$, so this angular error would be the most valuable data to analysis.

As shown in Fig. 3.1c, initially for each period, the robot one is more advanced than robot two, then since the torque output of robot one is always larger than robot two for each period, the rotational speed of phase of the more advanced one would be slowed down more by our controller, which leads to the synchronization eventually. This is the primary conclusion we can draw from the plot, but more detailed and theoretical analysis still needs to be carried out.

Essentially, all the data above can be derived from only two data: the target phase φ and the measured angle $\bar{\theta}$ along with their time index. In addition, thanks to the convenience of simulation, we can also record the data like contact force between the robots and the obstacle and positions of robots and obstacles to help us to better describe the dynamic model.

3.1.2 More findings on multiple robots

With the preliminary results of simply two robots, the ability for the robots to get synchronized has been confirmed, and the *mean resultant length* method has been applied to quantify the relationship between two robots. Then we move to study some more useful and practical cases, i.e. to implement the adaptive controller to more robots (5,10,and even 100 robots), so we can: (1) verify the consistency and effectiveness of our adaptive control method on different scales of

robot swarms; (2) get more generic results for us to understand the properties of the controller and to further optimize the parameters in our controller; (3) verify the advantages of synchronization compared with the non-sync case in the aspects of energy consumption, moving efficiency, etc.

First of all, the results for different choices of K_{adp} are obtained, which have been repeated for 5, 10, 20 and 50 robots respectively, as shown in Fig. 3.3. To make all the cases equivalent, the mass of the obstacle is set to be 50 g per robot, which offers 0.25 N friction as the load per robot. Since for the pushing case in the controller equations (2.6), the solution would stop rotating counterclockwise or even rotate backwards if the right hand side is not greater than zero, which would cause the robot to stop moving. In order to prevent this consequence from happening, the adaptive gain K_{adp} should be chosen carefully so that:

$$\omega_0 - K_{\text{adp}}(\bar{\theta} - \sqrt{\mu}\cos\varphi) > 0 \quad \forall(\bar{\theta} - \sqrt{\mu}\cos\varphi) \tag{3.4}$$

in order to get the maximum K_{adp} that can be chosen, and since during the pushing gait, $(\bar{\theta} - \sqrt{\mu}\cos\varphi)$ is always positive, we can derive the equation:

$$K_{\text{adp}} < \frac{\omega_0}{\bar{\theta} - \sqrt{\mu}\cos\varphi} \quad \forall(\bar{\theta} - \sqrt{\mu}\cos\varphi) \tag{3.5}$$

then the maximum value of adaptive gain is:

$$K_{\text{adp}}^{\max} = \frac{\omega_0}{\max(\bar{\theta} - \sqrt{\mu}\cos\varphi)} \tag{3.6}$$

For better understanding the synchronization results, we normalized the adaptive gain to be:

$$\gamma = \frac{K_{\text{adp}}}{K_{\text{adp}}^{\max}} = \frac{K_{\text{adp}}\max(\bar{\theta} - \sqrt{\mu}\cos\varphi)}{\omega_0} \tag{3.7}$$

in which $\max(\bar{\theta} - \sqrt{\mu}\cos\varphi)$ is the maximum value of the angular error. Then the normalized adaptive gain γ should be chosen less than one to ensure the robot's ability to move continuously. Thus, the test for γ larger than one could be ignored.

There are basically four aspects of results we can observed from the Fig. 3.3. Firstly, with the increasing in the number of robots, the trend of steady-state MRL value versus adaptive gain remains the same, which means the final synchronization status is largely depends on how

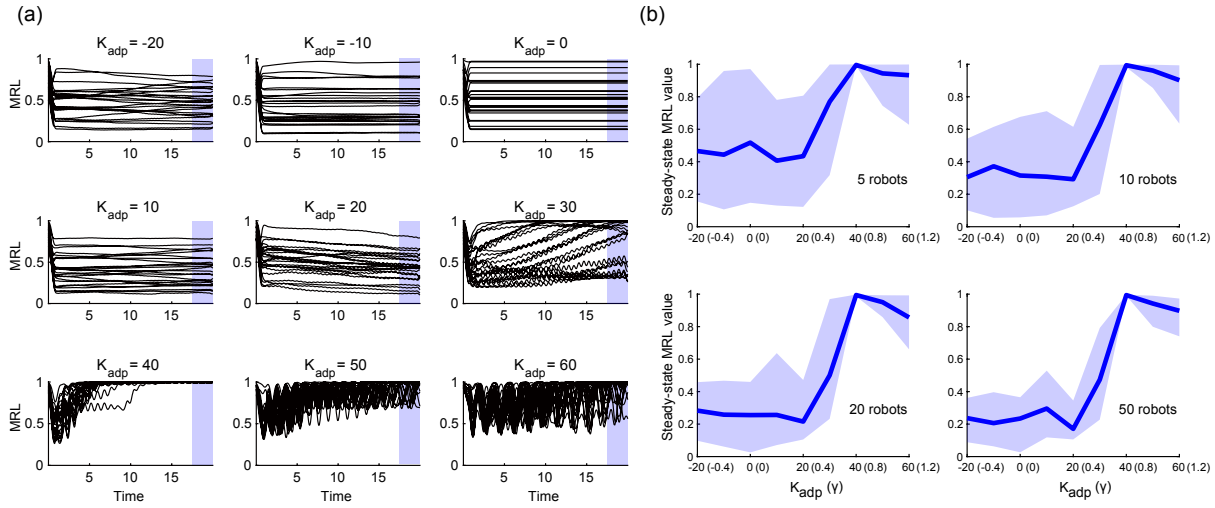

Figure 3.3: Simulation results with different K_{adp}. (a) Mean resultant length for 5 robots by choosing K_{adp} from -20 to 60, for each subplot there are 25 lines represent for 25 simulations with random initial conditions. (b) Steady-state mean resultant length values for different robots, for each subplots, the value is calculated by the last one period values from results like (a), e.g. for each blue region in (a), the mean value of 25 lines are calculated, then the mean, minimum and maximum value of these 25 values are obtained, finally a bounded line plot of the values versus different K_{adp} values can be made as the first subplot in (b).

much is the adaptive gain rather than the number of robots. And it suggests that the ability of K_{adp} to influence the collective behavior of robots remains the same in spite of how large the robot group is. Secondly, the optimal choice of adaptive gain for synchronization is around 0.8 of the $K_{\text{adp}}^{\text{max}}$. Thirdly, besides the results to be fully disordered eventually for K_{adp} less than zero, small K_{adp} values do not help with synchronization at all either. With observations of the graphics visualization rendered by simulation, small K_{adp} can not slow down the phase rotation enough, so the foot begins slides on the ground, which prevents synchronization. Lastly, the MRL value of being fully disordered is a value slightly above zero. This is because that the number of robot is finite, then the distribution of phases around the limit cycle would not be perfectly uniform, which cause the steady-state MRL from small and non-positive K_{adp} to be a little greater than zero. And the more the robots in the group, the more uniformly they distributed, the closer the MRL to zero.

Then, to figure out the external factors' influence on synchronization, simulations have

been done with different mass of the obstacle, which leads to different load to the robots, as shown in Fig. 3.4. Based on the previous results, the adaptive gain K_{adp} is 40 ($\gamma = 0.8$), which is optimal for synchronization.

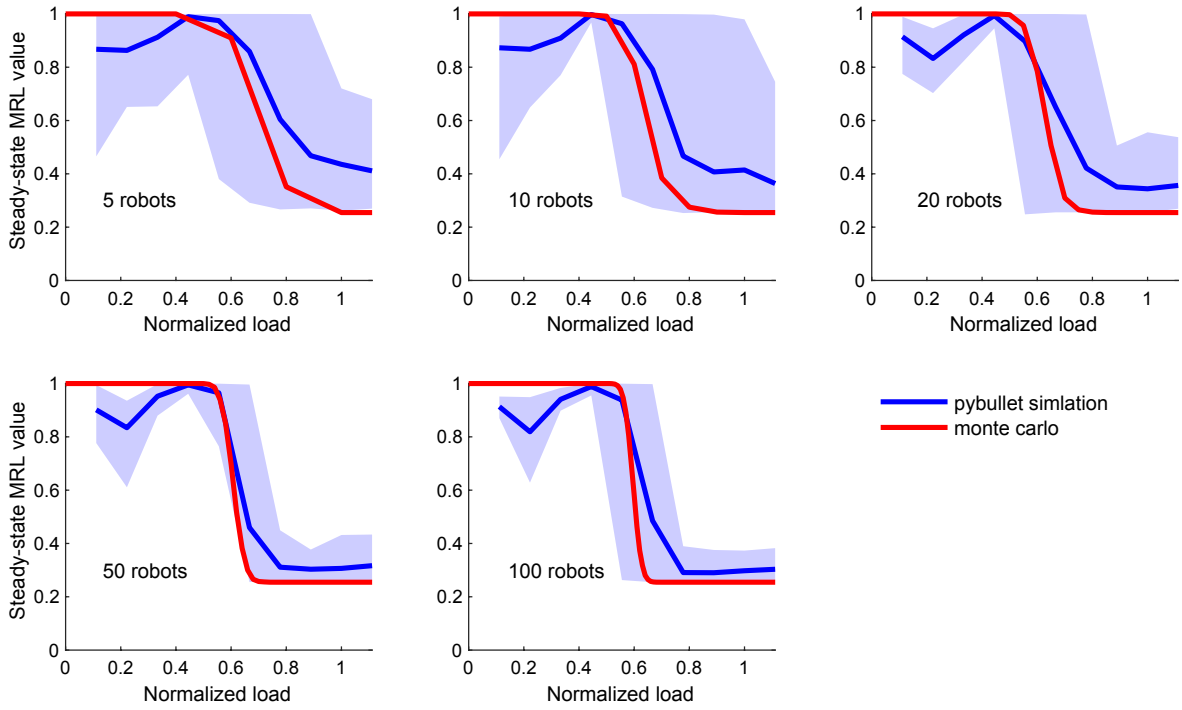

Figure 3.4: Results with different normalized load, where the load is normalized by the friction load offered by the obstacle divided by the force generated by each robot, then divided by the number of robots. The blue lines and regions are Pybullet simulation results, and the red lines are monte carlo simulation results as the theoretical results.

The normalized load is a parameter that 0 means no load and the 1 means the pushing force generated by all robots. When the normalized load is above one, in any case the robots are not able to push the obstacle forward, which means there are no interaction among all robots and they are random in phase. However, the results shows that the steady-state MRL value of completely disordered cases are not zero, but a value between 0.2 and 0.4. This result is cause by the large adaptive gain value so the the rotation speed is nonuniform around the limit cycle. Considering there are n robots with fully non-sync phases, the number of robots in the pushing gait and pulling gait could be n_1 and n_2 among all n robots. Equivalently, in this fully non-sync

systems, the pushing gait takes $\alpha = \frac{n_1}{n}$ of the entire time of one period, while the pulling gait takes $1-\alpha = \frac{n_2}{n}$. For each half period, the distribution of robots' phases can be regarded as uniform. Then from equation (3.1), we can get the MRL of each gait period:

$$R^2 = (\sum_{i=1}^{n_1+n_2} \cos\theta_i)^2 + (\sum_{i=1}^{n_1+n_2} \sin\theta_i)^2 \quad (R \geq 0), \quad \bar{R} = \frac{R}{n} \tag{3.8}$$

$$R_1^2 = (\sum_{i=1}^{n_1} \cos\theta_i)^2 + (\sum_{i=1}^{n_1} \sin\theta_i)^2 \quad (R_1 \geq 0), \quad \bar{R}_1 = \frac{R_1}{n_1} = \frac{R_1}{\alpha n} \tag{3.9}$$

$$R_2^2 = (\sum_{i=n_1+1}^{n_1+n_2} \cos\theta_i)^2 + (\sum_{i=n_1+1}^{n_1+n_2} \sin\theta_i)^2 \quad (R_2 \geq 0), \quad \bar{R}_2 = \frac{R_2}{n_2} = \frac{R_2}{(1-\alpha)n} \tag{3.10}$$

where $\bar{R}_1$ and $\bar{R}_2$ are mean resultant lengths of robots' phases in the pushing and pulling gait respectively, and $\bar{R}$ is the mean resultant length of all the phases. In this way, we are able to calculate the theoretical MRL value for completely non-sync robot swarms with the time proportion of pushing gait α, e.g. for the pushing gait half period:

$$\bar{R}_1 = \frac{\sqrt{(\sum_{i=1}^{\alpha n} \cos\frac{\pi i}{\alpha n})^2 + (\sum_{i=1}^{\alpha n} \sin\frac{\pi i}{\alpha n})^2}}{\alpha n} \tag{3.11}$$

furthermore, when we have massive number of robots, i.e. $n \to \infty$, we can derive:

$$\begin{aligned}
\bar{R}_1 &= \lim_{n\to\infty} \frac{\sqrt{(\sum_{i=1}^{\alpha n} \cos\frac{\pi i}{\alpha n})^2 + (\sum_{i=1}^{\alpha n} \sin\frac{\pi i}{\alpha n})^2}}{\alpha n} \\
&= \lim_{n\to\infty} \sqrt{\frac{\sum_{i=1}^{\alpha n}\sum_{j=1}^{\alpha n} \cos\frac{\pi i}{\alpha n}\cos\frac{\pi j}{\alpha n} + \sum_{i=1}^{\alpha n}\sum_{j=1}^{\alpha n} \sin\frac{\pi i}{\alpha n}\sin\frac{\pi j}{\alpha n}}{(\alpha n)^2}} \\
&= \sqrt{\int_0^1\int_0^1 \cos\frac{\pi}{\alpha n}\cos\frac{\pi}{\alpha n} \quad d\frac{1}{\alpha n} d\frac{1}{\alpha n} + \int_0^1\int_0^1 \sin\frac{\pi}{\alpha n}\sin\frac{\pi}{\alpha n} \quad d\frac{1}{\alpha n} d\frac{1}{\alpha n}} \\
&= \sqrt{\int_0^1\int_0^1 \cos\pi x\cos\pi y\, dx\, dy + \int_0^1\int_0^1 \sin\pi x\sin\pi y \quad dx\, dy} \\
&= \sqrt{0 + \frac{2}{\pi}\int_0^1 \sin\pi x \quad dx} \\
&= \frac{2}{\pi}
\end{aligned} \tag{3.12}$$

Similarly,

$$\bar{R}_2 = \frac{2}{\pi} \tag{3.13}$$

Since $\vec{R}_1$ and $\vec{R}_2$ have orientations on $\frac{\pi}{2}$ and $-\frac{\pi}{2}$, with the idea of vector addition, the MRL $\bar{R}$ of all phases can be calculated as:

$$\bar{R} = \frac{R}{n} = \frac{R_1 - R_2}{n} = \frac{\alpha n \bar{R}_1 - (1-\alpha) n \bar{R}_2}{n} = \alpha \bar{R}_1 - (1-\alpha)\bar{R}_2 = \frac{2(2\alpha - 1)}{\pi} \tag{3.14}$$

In this simulation, we have measured the proportion of the pushing gait time to the total time of one period as 0.7, with $\alpha = 0.7$, the theoretical MRL value of fully non-sync situation in this set of simulation is $\bar{R} = 2(2\alpha - 1)/\pi \approx 0.255$, which fits the results in Fig. 3.4.

From Fig. 3.4, the only two variables are the load and the initial conditions of robots. So we suppose that the how they can synchronized largely depends on whether there are enough robots to push the obstacle forward initially. To verify our hypothesis, we implement a monte carlo simulation to solve this probability problem.

To begin with, 10000 sets of random variables has been set for 5, 10,20,50 and 100 robots. As measured from the simulation, 70% of variables are uniformly distributed in $(0, \pi)$ and other 30% are uniformly distributed in $(-\pi, 0)$. Since the robots are not glued to the obstacle, there are always a distance for the robots to really contact the obstacle for each pushing gait, and the proportion of contact time to the total period is measured as about 0.4. Then for all number of robots, we count the cases that the certain number of robots is contacting the obstacle at the same time. For example for the red line in the first subplot of Fig. 3.4, the value of $x = 0.6$ is got by the proportion of the number of the cases that has $0.6 \times 5 = 3$ robots within $0.4 \times 2\pi = 0.8\pi$ to the all 10000 random cases. Finally, due to the fact that the lower and upper bound of MRL value in this simulation is 0.255 and 1, the values from monte carlo simulation are mapped to 0.255 to 1 to represent the final synchronization status of the system.

It can be observed from Fig. 3.4 that the results from monte carlo simulation are quite similar with the results from Pybullet simulation. Thus, our previous hypothesis, which supposes

the synchronization is largely depend on initial distributions of the robots' phases, has been verified.

3.1.3 Test of advantages from synchronization

Now we have detailed knowledge on how our control method affect the synchronization behavior among robots, and how we can properly set up the controller to better synchronize the robots. However, the benefits we can get from synchronization are not yet discovered. In other words, we should then manage to find out whether our adaptive controller and the synchronization behavior contribute to better performance of the robot group, and how better our method is compared with the non-sync group of robots.

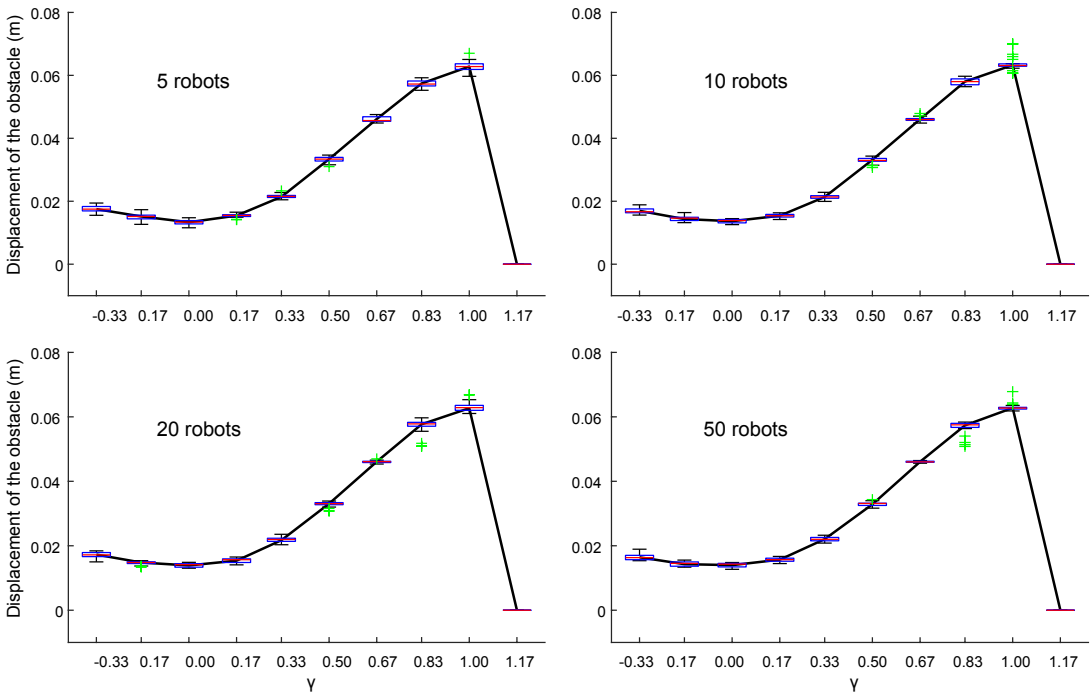

Figure 3.5: Comparisons of obstacle displacements with different γ in simulation.

To test the benefit we can get from synchronization, some comparisons have been made in Fig. 3.5, Fig. 3.6 and Fig. 3.7. In real application scenarios, the target of the robot would be to push a certain obstacle forward, so whether the robot is capable of pushing forward and how

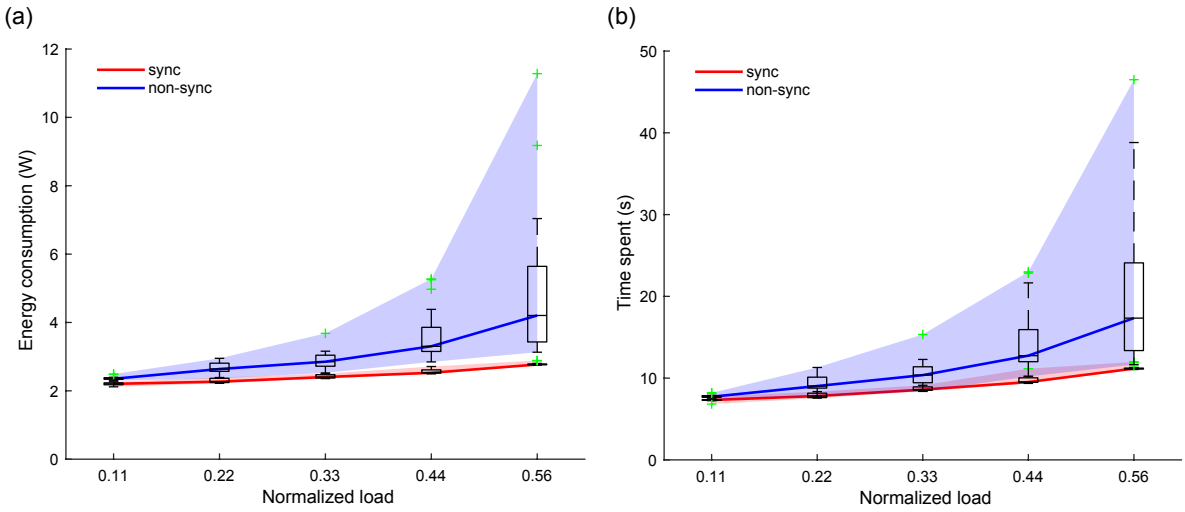

Figure 3.6: Comparisons of energy consumption, efficiency and pushing capacity between sync and non-sync.

efficiently the robot could finish the task are two indexes to be evaluated.

In Fig. 3.5, the displacements of obstacle with different γ are presented, which correspond to the plots in Fig. 3.3. In order to compare the results with equivalent circumstances, all the displacement data are captured just after the robots finished 10 moving cycles. As can be seen from the plots, the efficiency for robots to push the obstacle forward is low for small and negative γ ($\gamma < 0.4$), which corresponds to the non-sync cases as shown in Fig. 3.3. With the increasing of γ, the displacement for the robots to run 10 cycles significantly increases until $\gamma = 1$. So synchronization would definitely help the robots to improve the pushing efficiency. However, if $\gamma > 0$, the robot would stop moving against the obstacle, which is in agreement with the previous results.

In Fig. 3.6, 10 robots are arranged to push the obstacle for 0.1 m. For the x axis, the load is normalized as Fig. 3.4. Above normalized load of 0.56, the robots in the non-sync case are not able to push the obstacle forward anymore, so results above 0.56 are ignored. The sync case is set up with $\gamma = 0.8$, which is optimal for synchronization, while for the non-sync case, γ is 0 and ω_0 has been modified so that the actual frequencies of the leg are equivalent for sync and

non-sync cases. The energy consumption is approximated by the numerical integration of the output power, which equals to output toque multiplied by angular speed. As shown in the figures, with the increasing of the load, both energy consumption and total time spent increase, but for the non-sync case the values are always larger with more variance. All these results indicate that synchronization provide better working efficiency and consume less energy.

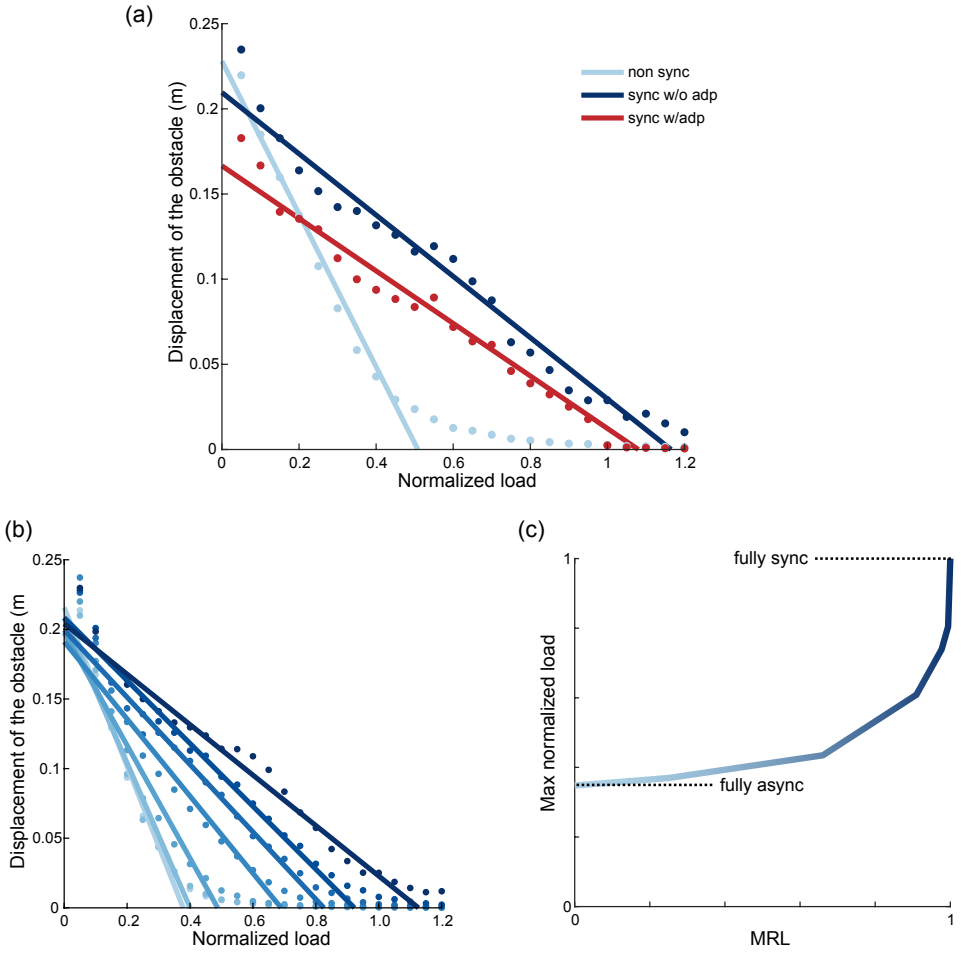

Figure 3.7: Comparisons of the pushing ability of the robots in simulation. (a) Comparison of obstacle displacements in non sync, sync and adaptive sync conditions. (b) Displacement of the obstacle in different sync conditions without adaptive gain (darker color indicates better sync). (c) Max pushing ability of robots calculated from x-intercept of (b).

The ability for the robots to push the obstacle forward is shown in Fig. 3.7a, 4 robots are used to push the obstacle, which are the same as what we have done in experiment. In this case,

all robots are allowed to run for 15 periods. There are two sync cases: sync with $\gamma = 0.8$ and disordered initial condition (in green) and sync with $\gamma = 0$ and same initial condition (in red) as a perfect sync case. And for the non-sync case, we set $\gamma = 0$ with disordered initial condition.

Fitted by the original data dots, the lines appears to be almost linear, which is basically caused by the existence of the static friction force of the obstacle, and the detailed reason could be explained by Fig. 3.8. There is always a error between the target angle $\sqrt{\mu}\cos\varphi$ and the measured angle $\bar{\theta}$ to generate the output torque (proportional to the output force). Then at the beginning of the pushing gait, the target angle start at $\bar{\theta} = \sqrt{\mu}$, at this time the robot is not able to push the obstacle forward until the target angle rotate to the position so that $\bar{\theta} - \sqrt{\mu}\cos\varphi$ is large enough to push the obstacle forward. This angular error can be regarded as a constant value during the entire pushing gait. And each pushing gait ends up with the target angle reaches $-\sqrt{\mu}$. With the previous derivation in equation (2.5):

$$\begin{aligned} & \theta_{\text{non-effective travel}} = \bar{\theta} - \sqrt{\mu}\cos\varphi \propto \tau \propto f = \mu_{\text{k}} M g \\ \text{s.t.} \quad & \theta_{\text{effective travel}} = \theta_{\text{total travel}} - \theta_{\text{non-effective travel}} \\ & \qquad = \theta_{\text{total travel}} - K_{\text{push}} M \end{aligned} \tag{3.15}$$

Where K_{push} is a constant coefficient, M is the mass of the obstacle, f is the static friction force of the obstacle and μ_{k} is the coefficient of friction. So the effective travel angle of one pushing gait is a linear function of M, which explains the linear fitted lines in Fig. 3.7a. Besides, these three lines show that robots can push almost three times as much as the weight of the obstacle by synchronization. This is because only half of the robots are pushing simultaneously when their phases are totally disordered. All other results with different γ and different initial conditions should be in between the light blue line (fully disordered) and the dark blue line (perfectly in phase).

Furthermore, to discover the relationship between pushing ability and synchronization status, simulations with varies initial conditions has been implemented (as shown in Fig. 3.7b and c). In this case, adaptive gain is set to be 0 so that robots would maintain their initial relationship

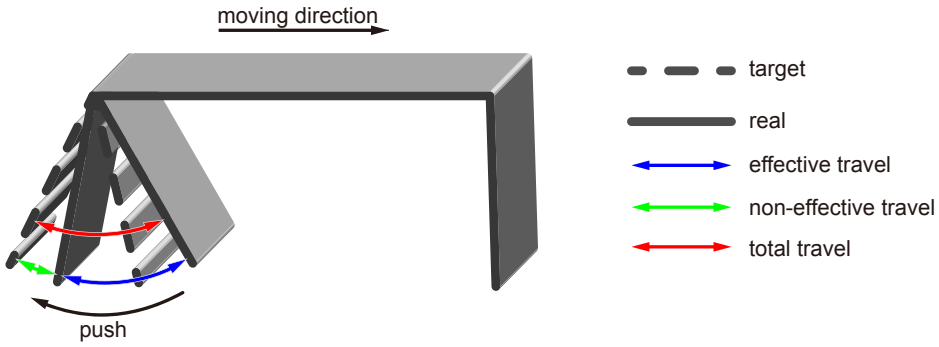

Figure 3.8: A schematic diagram about the movement decomposition of the pushing gait.

during the whole process. Fig. 3.7c is the x-intercepts of the fitted lines in Fig. 3.7b, it is clear to see that fully synchronous robots can generates three times pushing force together compared with the fully asynchronous robots. However, the robots can perform to have huge benefits only when they are perfectly synchronized or very close to fully synchronized, i.e. only when the MRL of robots is very close to 1 can the robots generate much more pushing force together compared with the non-sync condition.

Overall, with the help of simulations,a guideline for choosing the adaptive gain in the controller has been derived from the simulation. Some mathematical interpretations have been implemented to standardize the way we study the collective behavior of robot swarms. We also verified the advantages of the synchronization behavior realized by the adaptive controller in some certain application scenarios.

3.2 Experiment Results

Even though a great many of simulation results have been obtained, their effectiveness still remain to be verified. Then it comes up with the experiment. With the platform built for experiments, we set up 4 robots to push the obstacle together, which is a proper and enough amount to show the collective behavior of robots conveniently.

3.2.1 Some basic tests on four robots

The first set of experiments is to study the influence of the adaptive gain K_{adp} as we have done in Fig. 3.3. The mean resultant lengths of robots' phases are firstly calculated in Fig. 3.9a, which present 5 subplots with K_{adp} from -15 to 45. In each subplot, there are 6 lines represent for 6 experiments with random initial conditions. The revolution of MRL values as synchronization status can be easily observed from these subplots. As shown in Fig. 3.9b, the steady-state MRL values are calculated by the mean of last period MRL values in the blue regions of 3.9a.

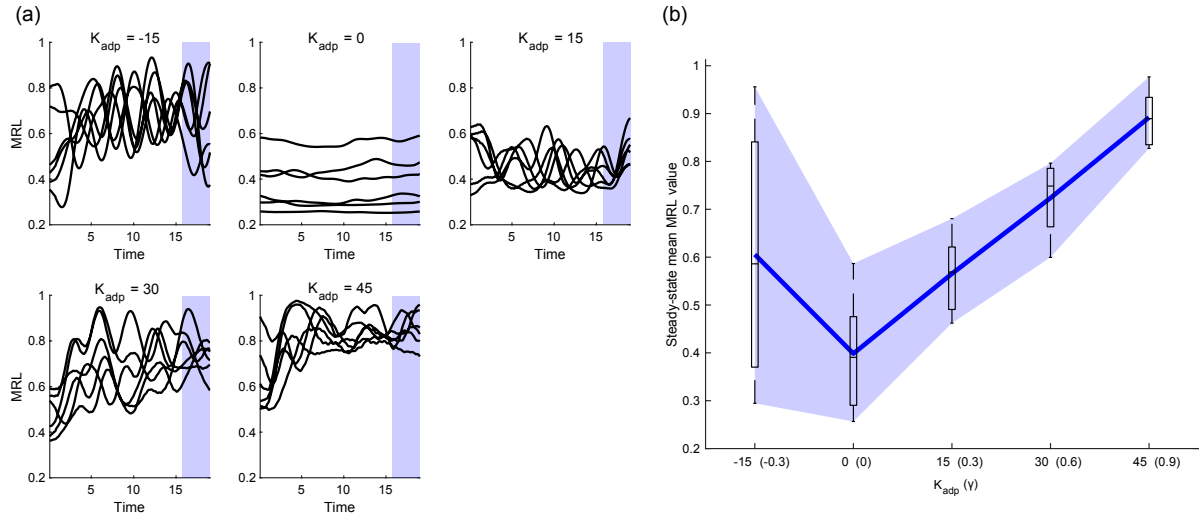

Figure 3.9: Experiment results with different K_{adp}. (a) MRL for 5 robots by choosing K_{adp} from -20 to 60, for each subplot there are 25 lines represent for 25 simulations with random initial conditions. (b) Steady-state mean resultant length values for different robots, for each subplots, the value is calculated by the last one period values from results like (a), e.g. for each blue region in (a), the mean value of 25 lines are calculated, then the mean, minimum and maximum value of these 25 values are obtained, finally a bounded line plot of the values versus different K_{adp} values can be made as the first subplot in (b).

The first finding of this figure is the range of the MRL values, which are between 0.2 and 1. Recall that $\omega_0 = 15\,\text{rad/s}$ and $\omega_1 = 30\,\text{rad/s}$. With the theoretical explanation in equation (3.14), α should be $\frac{2}{3}$, and for $\gamma = -0.3$, the pushing gait is accelerate a little bit, which leads the minimum α in our experiments to be around 0.66. So the theoretical minimum value of MRL for the fully non-sync case is $\bar{R} = \frac{2(2\alpha-1)}{\pi} \approx 0.20$, which is consistent with the minimum value

of MRL in Fig. 3.9a. Another result is about the influence of γ on synchronization. Fig. 3.9b provides the final status of the robots w.r.t γ from -0.3 to 0.9. Larger γ values help robots to synchronize better with each other, while small γ values would not contribute to synchronization, and negative γ values would even drive the robots to a totally disordered behavior. The cases of $\gamma > 1$ are not presented because robots would stop moving forever, as equation (3.7). In spite of the noisy data with more variance caused by the imperfect experiment conditions, the overall results and rules of this set of experiments are consistent with the experiment results in Fig. 3.3.

3.2.2 Test of advantages from synchronization

Then we manage to verify the effectiveness of synchronization behavior in the aspect of efficiency and ability to push the obstacle forward, as shown in Fig. 3.10 and Fig. 3.11. These experiments exactly correspond to the previous simulations, except for the results related to energy consumption since it is hard to measure the input or output power precisely with the current robot hardware.

Fig. 3.10a provides a visualized view of the obstacle displacements. In this experiment, the mass of the obstacle is 180 g, and all the robots are allowed to run for 25 periods. It can be seen from the overhead photo that higher γ value can better improve the efficiency of the robot's pushing ability. However excessive value of $\gamma > 1$ would stall all the motions. Thanks to the computer vision toolbox in MATLAB, the displacements can be measured from the experiment videos. In this way, a box plot has been generated as Fig. 3.10b. For each box, there are 6 experiments with random initial conditions. The above overhead photo exactly shows a typical case for each box, as they shows the similar arrangement of results. With the box plot presented, we can better compare this experiment result with the simulation results in Fig. 3.5. This result appears to be highly consistent with the simulation result, which again confirmed the positive influence of the adaptive controller and synchronization behavior on pushing efficiency of robot swarms.

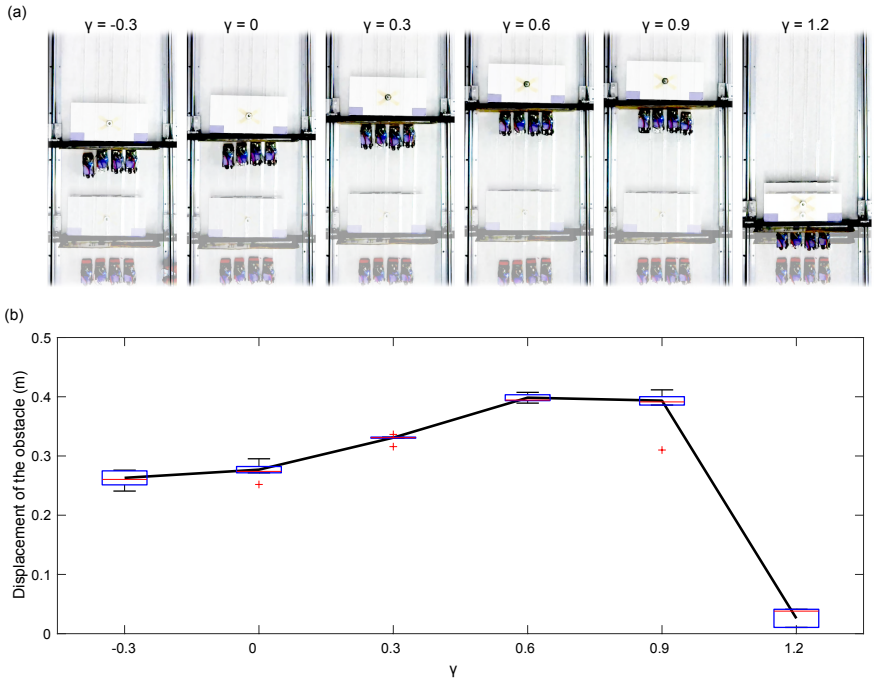

Figure 3.10: Comparisons of obstacle displacements with different γ in experiment. (a) An overhead view of the beginning positions (transparent) and the end positions (solid). (b) The displacement results data for different γ

Fig. 3.11 shows how our adaptive controller and synchronization affect how much weight that the robots are able to push forward. All the robots in this experiment are allowed to run for 15 periods and we would measure the displacements of the obstacle from the recorded videos as the previous experiment. This experiment is for comparison between different cases, so it is unnecessary to measure the friction coefficient and the output force of one robot to get the normalized load (as in simulation), we simply use the total weight of the obstacle as the x axis. For the non-sync case, robots are set up with disordered initial conditions and $\gamma = 0$. For the sync with adaptive case, robots are also set up with disordered initial conditions but γ is 0.8 to let the robot synchronize with each other. And for the sync without adaptive case, all the robots have the same initial conditions and $\gamma = 0$, so that all the robots keep walking perfectly in phase. This result has a lot of similarities with the simulation result in Fig. 3.6c. Firstly, the synchronized robots can provide twice as much as the load of non-sync ones. Then, the fitted lines

of displacement versus the weight are almost linear, and the reason has already been discussed in equation (3.15). Thus, the advantage of synchronization has been validated once again.

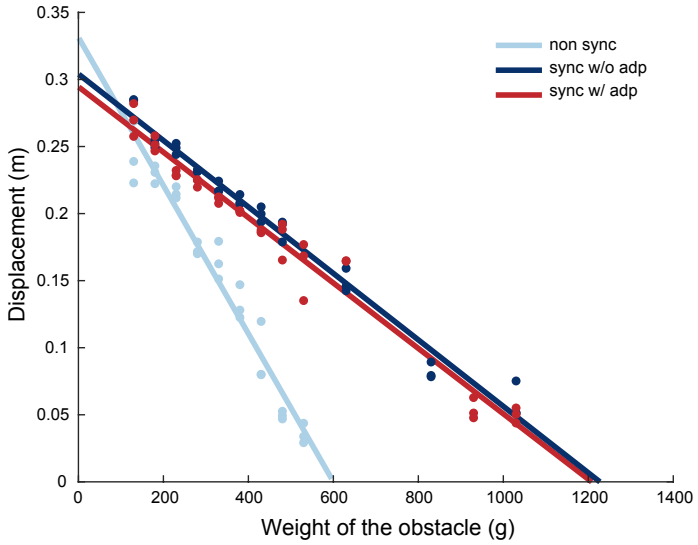

Figure 3.11: Comparisons of the pushing ability of the robots in experiment.

From all the experiment results above, synchronization has been proved to have benefits on improving the the robot group's efficiency and the ability to pushing against loads. In real experiment, due to the imperfect conditions, the adaptive may not strictly and perfectly make the robots to synchronize with each other. However, as a result, even with imperfect synchronization behavior, the performance of the robots still increased a lot, since the adaptive controller is able to let all the robots push simultaneously rather than generate intermittent pushing force to the obstacle.

3.2.3 Other miscellaneous experiments

The previous pushing-against-obstacle experiment is just one of varies possible application scenarios. With the existing experiment set up, there are a great many possible combinations for the robots to finish certain tasks. As shown in Fig. 3.12, two possible working scenarios are presented.

In Fig. 3.12a, two robots are pushing the obstacle in parallel, while the other two robots are pushing behind the two robots in front. The synchronization behavior not only occurs between the two robots in parallel, but also between the front one and the rear one in series. In this way, the number of robots we can set for pushing the obstacle would not be limited with the width of the obstacle, since we can place robot in series and the synchronization among all robots in the group can coordinate to generate huge forces simultaneously.

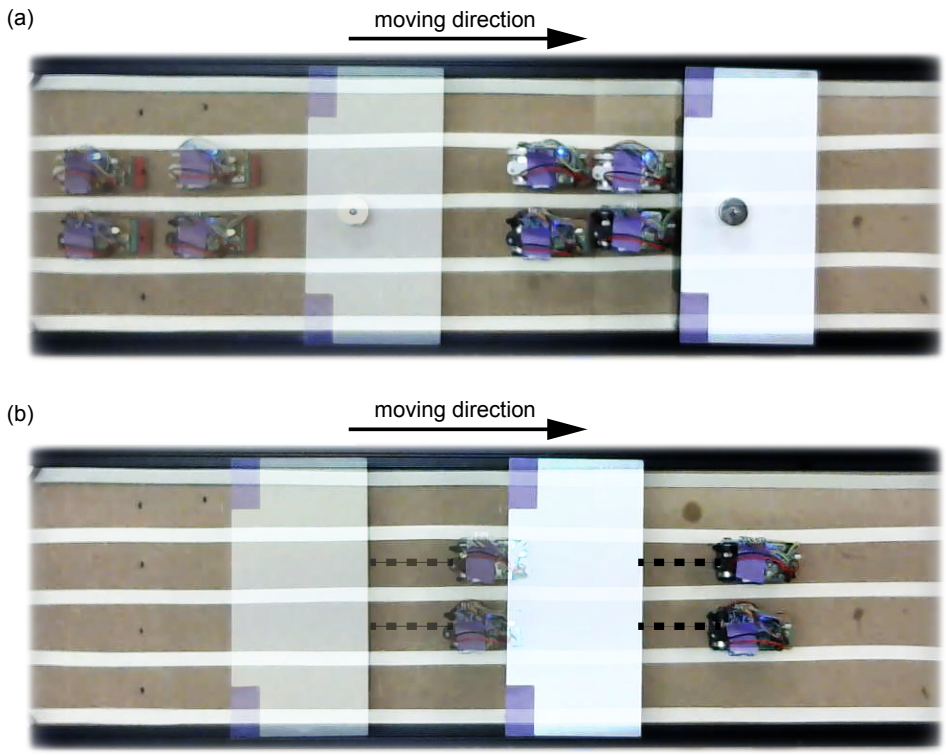

Figure 3.12: Other two types of application scenarios. (a) Four robots pushing in parallel and in series. (b) Two robots pulling with strings.

Another example is shown in Fig. 3.12b. The robots in this case no loner push the obstacle forward. Instead, they are pulling the obstacle together with strings (for better visibility the strings are shown in dotted lines). In this situation, the robots are connected with the obstacle with a soft string, so the pulling force only generates during the pushing gait of each robot. Even though pushing and pulling the obstacle appears to be totally different cases, the mechanisms behind

them are exactly the same. Through the force interactions during the pushing gait between these two robots, synchronization behavior has been observed as well. With the help of synchronization, these two robots can drag the obstacle forward, which is too heavy for a single robot.

Moreover, inspired by the synchronization of metronomes [34] and the crowd synchrony on the Millennium Bridge [35], we set up another experiment platform that floats above two polished iron rods (as shown in Fig. 3.13).

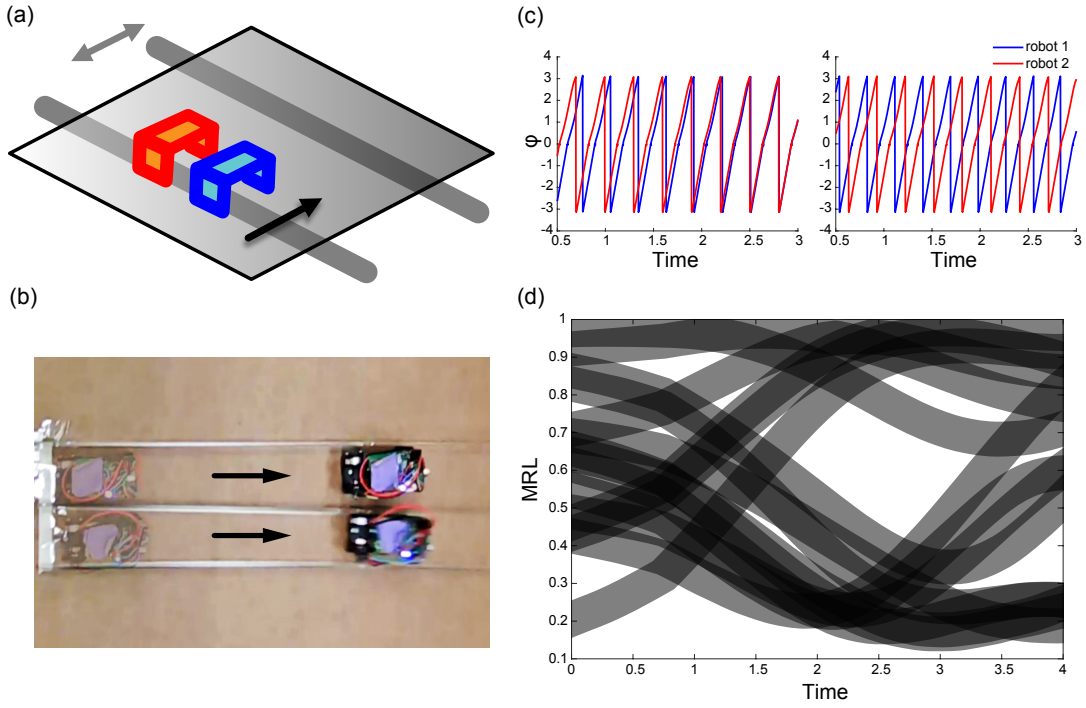

Figure 3.13: Robots walking on a floating platform. (a) A schematic of floating platform. (b) A top view of the experiment. (c) Two types of results: in phase and anti-phase. (d) The MRL revolution of the whole process from 15 random experiments.

Some basic test has been done with two robots walking on this floating platform (the screen shot of the recorded video from the overhead view is shown in Fig. 3.13b). Still controlled by out adaptive controller, With right set of parameters, these two robots exhibit some regular behaviors. First of all, unlike the previous experiments that requires positive adaptive gain K_{adp} for synchronization, now we observed that the robots only synchronized with negative adaptive gains. Rather than synchronized by slowing down against pushing the obstacle, these two robots with

negative adaptive gains tend to accelerate and amplify the motion of the floating platform, then the motion of the platform would speed up the slower robot until all robots walk in phase. However, synchronization is not guaranteed in this case. We also observed some anti-phase behaviors (a typical synchronization behavior is shown in Fig. 3.13c left, and a anti-phase behavior is shown in Fig. 3.13c right), which happen in the metronomes experiment as well. With some particular anti-phase initial conditions, two robots may get into the pushing gait consequently, which means one robot is in the pushing gait while another one is in the pulling gait. The forces generated by robots act in the opposite direction and counteract with each other, so the platform hardly oscillates and the robots remain anti-phase. The revolution of the mean resultant length of phases from 15 random experiments are shown in Fig. 3.13d, where almost all robots end up with the either in phase synchronization ($MRL = 1$) or anti-phase ($MRL \approx 0$). Moreover, this anti-phase behavior would be uncommon with the increasing of the number of robots.

Chapter 4

Synchronization Analysis

Now we have massive primary data from both simulations and experiments, some preliminary discussions and analyses have been carried out in the last chapter. In this chapter, we aim to find more detailed information from results data, further verify and discover the synchronization property of our robot system, and explain the basic logic behind synchronization behavior. To simplify the analysis and give a clear clarification, we only focus on the interactions between two robots.

4.1 Return maps

Involved with collisions, frictions and other force interactions, even though the controller equations are well defined, the whole robot system is still complex and non-linear. Therefore, we must find another way to simply express the synchronization behavior of our robot system. According to the periodic behavior of the phase φ in the adaptive controller, the interactions and synchronization behaviors are also periodic. By ignoring the intermediate process in each period, a discrete expression can be derived as:

$$\Delta^n \rightarrow \Delta^{n+1} \tag{4.1}$$

where Δ^n is typically defined as the phase difference $|\varphi_1 - \varphi_2|$ when all the two robots are in the pulling gait. And the superscript n stands for the n^{th} period from the beginning. By reasons of the changing of the phase angular speed around the limit cycle, a phase difference should be captured from the equivalent place of each period to represent the relationship between two robots. Thus, we typically choose Δ^n when all the two robots are in the pulling gait, since it has the constant phase speed ω_1.

Some return maps of Δ from both simulation and experiment results have been presented in Fig. 4.1. In the two robot case, the phase evolution for each robot should be equivalent, which means the sign of the subtraction between robot 1's phase and robot 2's phase does not matter. Then the phase difference Δ is an absolute value between 0 and π.

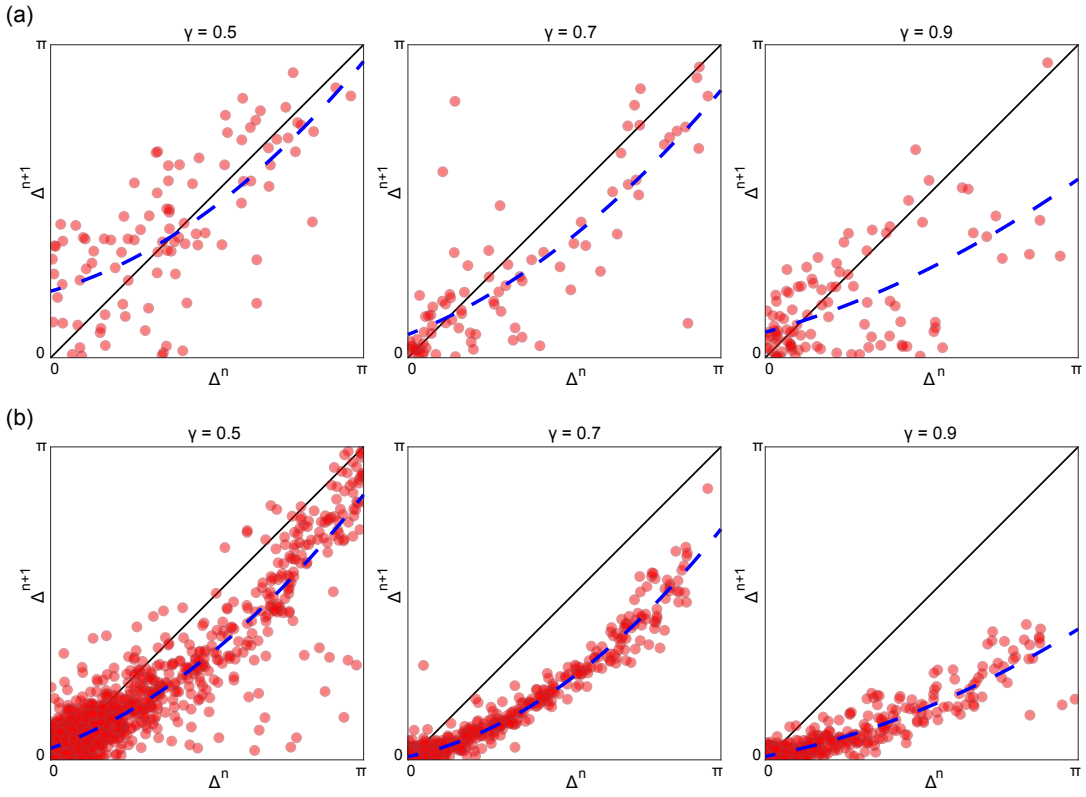

Figure 4.1: Return map of two robots' phase difference. (a) Results from experiments. (b) Results from simulations. (raw data are shown in red dots, fitted curves are shown in blue dotted lines)

Despite that the experiment data are quite scattered compared with the data from the

simulation, the fitted curve shows great similarity for each case. With the knowledge of our previous results, if the normalized adaptive gain is chosen to be higher than 0.5 but less than 1, then we can observe synchronization phenomenon among robot swarms. Besides, the higher adaptive gain is, the better the robots get synchronized. In the return maps, most dots distribute lower than the $\Delta^n = \Delta^{n+1}$ line, especially for the cases with $\gamma = 0.7$ and $\gamma = 0.9$. This indicates the phase difference would finally converge to zero, so the two robots get synchronized. The fitted lines illustrate the converge rate, with larger γ, the phase difference would converge to zero faster, which again verify the results in Chapter 3. Note that for experiments, there are certain amount of data dots above the $\Delta^n = \Delta^{n+1}$ line, particularly for small Δ values, even for large normalized adaptive gains like $\gamma = 0.9$. That is because there are plenty of imperfect factors in the experiment act as perturbations, so we must apply high adaptive gains to ensure the converge of synchronization in practice.

4.2 Contact patterns

With the basic guarantee of synchronization from the return maps, now we manage to break the whole process down to small cases, simply regarding whether the robots are contacting the obstacle (as shown in Fig. 4.2).

Generally speaking, the process for two robots to interact with the obstacle alternately can be described as: (a) Firstly One robot is in contact with the obstacle (Fig. 4.2a): the front robot (in this case is the robot 1) firstly contact with the obstacle, while there is a still distance for the other one to contact. (b) Secondly two robots are in contact with the obstacle together(Fig. 4.2b): the behind robot (robot 2) catch up with the front one, then two robots push the obstacle forward together. (c) Then just another robot is in contact (Fig. 4.2c): the front robot (robot 1) enters the pulling gait so it is not in contact, while the other one is still in contact with the obstacle. (d) Finally no robot is in contact (Fig. 4.2d): two robots are all in the pulling gait. Afterwards

these two robots would repeat the process from (a) to (d) over and over again until they get fully synchronized.

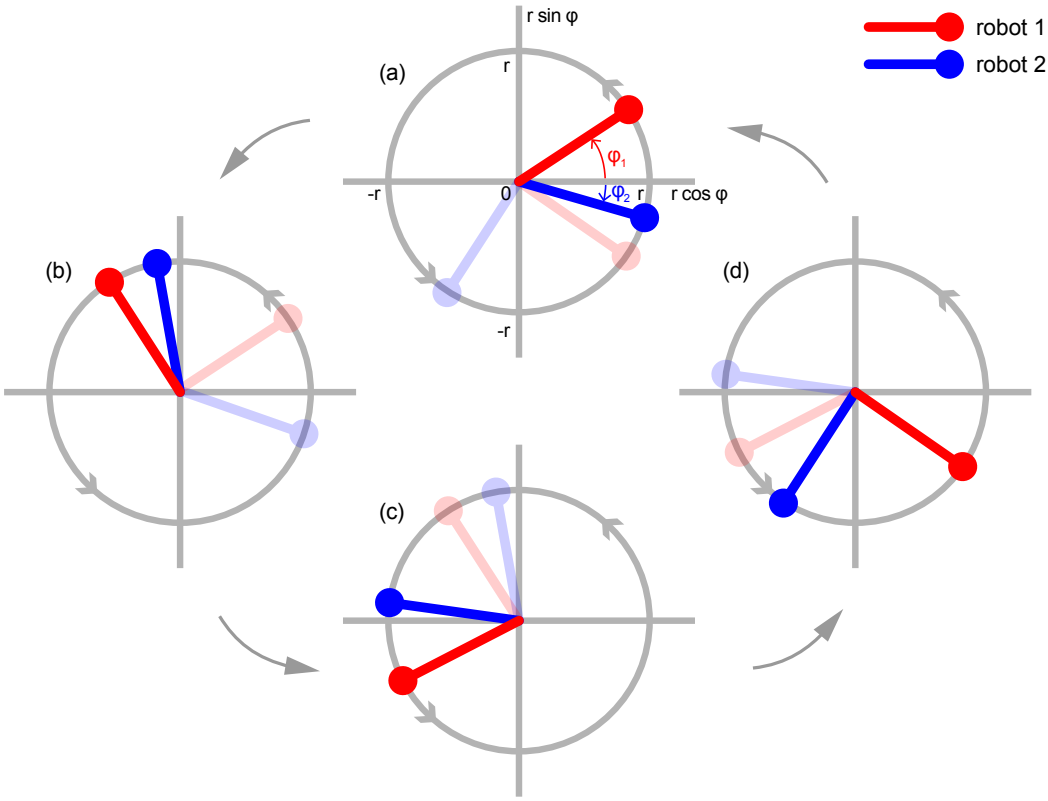

Figure 4.2: Contact patterns of two robots. (a) Robot 1 in contact with the obstacle. (b) Two robots all in contact. (c) Robot 2 in contact. (d) No robot in contact.

These four cases above can be classified into three contact patterns : one in contact; two in contact and no in contact. To understand the detailed synchronization process in each contact patterns, we collect data from hundreds of simulations and experiment with $\gamma = 0.9$, then calculate the average change rates of the the phase difference Δ, which are shown in Fig. 4.3.

It can be seen from the data in Fig. 4.3 that almost all the change of phase difference occurs during the "one in contact" pattern, and for the other two patterns, the phase difference almost does not change. Since the "one in contact" pattern is the only case that two robots yield different dynamic situations (in the other two cases, they are both impeded by the obstacle or both move freely), and synchronization process only takes place during this pattern. We may

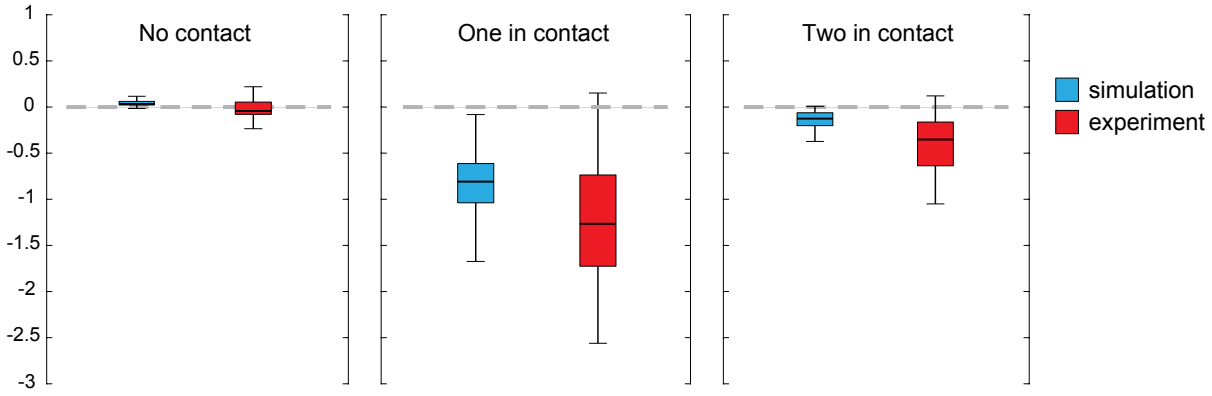

Figure 4.3: Box plots of the average phase difference change rates in three contact patterns.

consider our controller as a "contact detector", and this detector works perfectly to identify the different contact patterns above to realize the final sync behavior between two robots.

4.3 Simplified theoretical model

Back to our adaptive controller equations (2.6), the phase φ of each robot is a periodic value affected by external force perturbations. Since the phase data is the only data that we can directly and easily recorded from the controller, and it is smooth compared with the angular data measured from the motor encoder, we use it to represent for the status of each robot in the previous analyses.

In order to theoretically analyse the synchronization behavior of the system, we assume that the force is constant when the robot is pushing itself or pushing against the obstacle. In fact, the measured angle $\bar{\theta}$ is oscillating around a positive equilibrium since the weight of the robot itself always apply torque to drive the leg angle towards the positive direction. So we simply ignore the transformation process from one contact case to another, and with the idea from the last subsection, we define the angular error in these two cases as c_1 and c_2, in which $c_1 < c_2$ (e.g. from pulling gait to pushing itself, $\bar{\theta}$ instantly move from $\sqrt{\mu}$ to $\sqrt{\mu} + c_1$ while ϕ remains 0 at

this point). Hence, the controller equations are ($\dot{\varphi}$ always greater than 0):

$$\dot{\varphi} = \begin{cases} \omega_0 - K_{\text{adp}}\, c_1, & \text{for pushing itself} \\ \omega_0 - K_{\text{adp}}\, c_2, & \text{for pushing the obstacle} \\ \omega_1, & \text{for pulling gait: } \sin\varphi < 0 \end{cases} \tag{4.2}$$

Firstly we consider the case that two robots are far away from each other, robot 1 is close to the obstacle and contacting the obstacle to move it forward, but robot 2 still needs time to move close to the obstacle. In this simple case, we consider the distance for each robot to move one period is the same. The time for both robots to spend in the pulling case is the same as: $t_{\text{pull}} = \frac{\pi}{\omega_1}$. However, for the pushing gait, robot 1 spends $t_{\text{push1}} = \frac{\pi}{\omega_0 - K_{\text{adp}}\, c_2}$ as it pushes against the obstacle; robot 2 spends $t_{\text{push2}} = \frac{\pi}{\omega_0 - K_{\text{adp}}\, c_1}$ as it just pushes itself forward. Since $c_1 < c_2$, then $t_{\text{pull}} + t_{\text{push1}} > t_{\text{pull}} + t_{\text{push2}}$, which means robot 1 needs more time to run one period. Thus, robot 2 moves faster than robot 1 and would eventually catch up with robot 1, then two robots would contact together with the obstacle.

Then we consider the case that two robot contact together with the obstacle. These two robots start at the initial phase state $[\varphi_1^0, \varphi_2^0] = [0, -\alpha]$, so robot 1 is the leading robot with advanced phase. Strictly speaking, perfect synchronization not only means sync in phase, but also the position of each robot. Here we introduce the position difference δ along with the phase difference Δ. Now we have the full state $[\Delta, \delta] = [\alpha, \sigma]$ to represent the complete relationship between 2 robots. In order to analyse the detailed process, some typical cases are shown in Fig. 4.4.

The starting phases are shown in Fig. 4.4b, as robot 2's phase is α behind of robot 2, and robot 1 now is σ away from the obstacle. Backward to Fig.4.4a, the σ is caused by the robot 2 pushing forward while robot 1 is transferring from pushing gait to pulling gait. So σ can be directly represented by $\frac{\sigma}{l} = r(1 - \cos\beta)$, where l is the length of the leg and β is shown in Fig.4.4a. When robot 2 enters the pulling gait, robot 1's phase should be $-\pi + \alpha$. According to equation (4.2), the

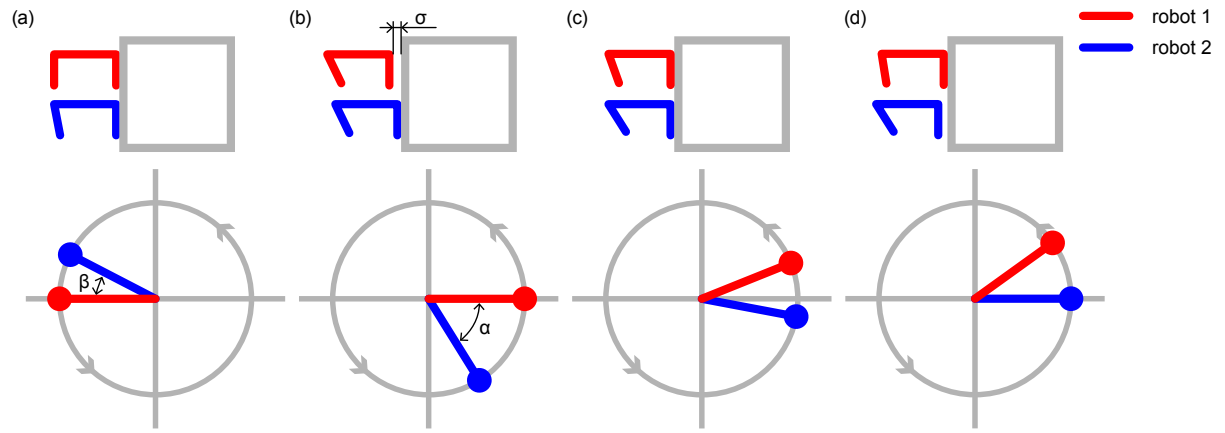

Figure 4.4: Four typical cases of two robots. (a) Robot 1 is entering the pulling gait while robot 2 is in the pushing gait. (b) All the robots are in the pulling gait. (c) Robot 1 is in the pushing gait while robot 2 is in the pulling gait. (d) Robot 1 is pushing the obstacle while robot 2 is entering the pushing gait but not touching the obstacle.

time for robot 2 move from Fig.4.4a to pulling gait is $t = \frac{\beta}{\omega_0 - K_{\text{adp}}\, c_2} = \frac{\alpha}{\omega_1} \Rightarrow \beta = \frac{\alpha\left(\omega_0 - K_{\text{adp}}\, c_2\right)}{\omega_1}$. So $\frac{\sigma}{l} = r(1 - \cos\frac{\alpha\left(\omega_0 - K_{\text{adp}}\, c_2\right)}{\omega_1})$. This result shows the position difference δ is actually related to the phase difference Δ, so we only need to prove either one to prove the synchronization for both of them. The detailed analysis is shown below:

1. From Fig.4.4b to Fig.4.4c (robot 1 entering pushing gait to robot 1 contacting with the obstacle):

$$\begin{aligned} \text{robot 1:} \qquad \varphi_1^1 &= \cos^{-1}\left(r - \frac{\sigma}{l}\right) \\ &= \frac{\alpha\left(\omega_0 - K_{\text{adp}}\, c_2\right)}{\omega_1} \\ t_1^1 &= \frac{\varphi_1^1 - \varphi_1^0}{\omega_0 - K_{\text{adp}}\, c_1} \\ &= \frac{\alpha\left(\omega_0 - K_{\text{adp}} c_2\right)}{\omega_1\left(\omega_0 - K_{\text{adp}}\, c_1\right)} \end{aligned} \tag{4.3}$$

$$\begin{aligned} \text{robot 2:} \qquad \varphi_2^1 &= -\alpha + t_1^1 \omega_1 \\ &= \frac{\alpha K_{\text{adp}}\ (c_1 - c_2)}{\omega_0 - K_{\text{adp}}\, c_1} < 0 \end{aligned} \tag{4.4}$$

2. From Fig.4.4c to the next case like From Fig.4.4a (robot 1 contacting with the obstacle to

robot 1 entering the pulling gait):

(a) Suppose $(c_2 - c_1)$ is very small that robot 1 is not able to catch up to contact with the obstacle:

$$\begin{aligned}
\text{robot 1 goes to } \pi: \qquad \varphi_1^2 &= \pi \\
t_1^2 &= \frac{\varphi_1^2 - \varphi_1^1}{\omega_0 - K_{\text{adp}}\, c_2} \\
&= \frac{\pi}{\omega_0 - K_{\text{adp}}\, C_2} - \frac{\alpha}{\omega_1}
\end{aligned} \tag{4.5}$$

$$\begin{aligned}
\text{robot 2 goes to 0:} \qquad \varphi_2^{2_1} &= 0 \\
t_2^{2_1} &= \frac{\varphi_2^{2_1} - \varphi_2^1}{\omega_1} \\
&= \frac{\alpha K_{\text{adp}}\ (c_2 - c_1)}{\left(\omega_0 - K_{\text{adp}}\, c_1\right) \omega_1}
\end{aligned} \tag{4.6}$$

$$\begin{aligned}
\text{robot 2 from 0 to the end:} \qquad t_{2_2}^2 &= t_1^2 - t_2^{2_1} \\
&= \frac{\pi}{\omega_0 - K_{\text{adp}}\, c_2} + \frac{\alpha \left(2K_{\text{adp}}\, c_1 - \omega_0 - K_{\text{adp}}\, c_2\right)}{\omega_1 \left(\omega_0 - K_{\text{adp}}\, c_1\right)} \\
\varphi_2^{2_2} &= \varphi_2^{2_1} + t_2^{2-2} \left(\omega_0 - K_{\text{adp}}\, c_1\right) \\
&= \frac{\pi \left(\omega_0 - K_{\text{adp}}\, c_1\right)}{\omega_0 - K_{\text{adp}}\, c_2} + \frac{\alpha \left(2K_{\text{adp}}\, c_1 - \omega_0 - K_{\text{adp}}\, c_2\right)}{\omega_1}
\end{aligned} \tag{4.7}$$

Recall that

$$\beta = \frac{\alpha \left(\omega_0 - K_{\text{adp}}\, c_2\right)}{\omega_1} \tag{4.8}$$

Now we have the new value of β after one period

$$\beta^+ = \pi - \varphi_2^{2\_2} = \frac{\pi K_{\text{adp}}\ (c_1 - c_2)}{\omega_0 - K_{\text{adp}}\, c_2} + \frac{\alpha \left(\omega_0 + K_{\text{adp}}\, c_2 - 2K_{\text{adp}}\, c_1\right)}{\omega_1} \tag{4.9}$$

Compare this new β^+ with the previous β we get

$$\beta^+ - \beta = \frac{K_{\text{adp}}\ (c_1 - c_2) \left[\omega_1 \pi - 2\alpha \left(\omega_0 - K_{\text{adp}}\, c_2\right)\right]}{\omega_1 \left(\omega_0 - K_{\text{adp}}\, c_2\right)} \tag{4.10}$$

According to the previous definition, $c_1 - c_2 < 0$, $\omega_0 - K_{\text{adp}}\, c_2 > 0$. Since in practice

we always choose ω_1 much greater than ω_0, and with large K_{adp} , $\omega_0 - K_{\text{adp}}\, c_2$ is a very small value, so we can consider $\omega_1 >> \omega_0 - K_{\text{adp}}\, c_2$. Along with $\pi > \alpha$, we can derive $\left[\omega_1 \pi - 2\alpha\left(\omega_0 - K_{\text{adp}}\, c_2\right)\right] > 0$. From the inequalities above, finally we conclude that $\beta^+ - \beta < 0$. Therefore, the position difference would decrease from period to period and eventually synchronized, so does the phase difference. However, in the previous Fig.4.4a case, two robots already contact the obstacle at the same time, with the results above, these robots would gradually synchronize with each other. So in the next Fig.4.4a case, these two robots are only possible to also contact with the obstacle simultaneously, which contradicts with our assumption from the beginning. Thus, the two robots must already start to contact the obstacle simultaneously from Fig.4.4d to Fig.4.4a.

(b) Robot 1 and robot 2 start to contact the obstacle together at some time from Fig.4.4d to Fig.4.4a: φ_2 would approach φ_2 until they both contact with the obstacle. Suppose that at this time, the difference of measured angle between two robots, which is $\frac{\sigma}{l}$, has not changed from the beginning. When these two robots both contact with the obstacle, the angular errors $\bar{\theta} - r\cos\varphi$ are both defined to be c_2, so we assume that $r(\cos\varphi_2^1 - \cos\varphi_1^1) = \frac{\sigma}{l}$ when two robots just contact with the obstacle together. At this time, we define $\varphi_1^1 - \varphi_2^1 = \varepsilon$ to meet $r(\cos\varphi_2^1 - \cos\varphi_1^1) = \frac{\sigma}{l}$. And we also assume that $\varphi_1 - \varphi_2$ would keep the difference of ε until the next Fig.4.4a. We can simply analyse the two typical time points when robots start to contact the obstacle together (defined as φ_1^1 and $\varphi_2^1 = \varphi_1^1 - \varepsilon$) and robot 1 just entering the pulling gait (defined as $\varphi_1^2 = \pi$ and $\varphi_2^2 = \pi - \varepsilon$). The new position difference could be defined as:

$$\frac{\sigma^+}{l} = r[\cos(\pi - \varepsilon) - \cos\pi] \tag{4.11}$$

Then we compare the new σ^+ with the previous σ

$$\begin{aligned}\frac{\sigma^+}{l}-\frac{\sigma}{l} &= r[\cos(\pi-\varepsilon)-\cos\pi]-r[\cos(\varphi_1^1-\varepsilon)-\cos\varphi_1^1] \\ &= r\left(-2\sin\frac{2\pi-\varepsilon}{2}\sin\frac{-\varepsilon}{2}+2\sin\frac{2\varphi_1^1-\varepsilon}{2}\sin\frac{-\varepsilon}{2}\right) \\ &= 2r\sin\frac{\varepsilon}{2}\left(\sin\frac{\varepsilon}{2}-\sin\frac{2\varphi_1^1-\varepsilon}{2}\right) \\ &= 4r\sin\frac{\varepsilon}{2}\cos\frac{\varphi_1^1}{2}\sin\frac{\varepsilon-\varphi_1}{2}\end{aligned} \tag{4.12}$$

Since

$$\begin{aligned}&0<\varepsilon<\pi, \quad 0<\varphi_1^1<pi, \quad \varepsilon-\varphi_1^1=-\varphi_2^1<0 \\ \Rightarrow \quad &\sin\frac{\varepsilon}{2}>0, \quad \cos\frac{\varphi_1^1}{2}>0, \quad \sin\frac{\varepsilon-\varphi_1}{2}<0\end{aligned} \tag{4.13}$$

Then

$$\frac{\sigma^+}{l}-\frac{\sigma}{l}<0 \quad \Rightarrow \quad \sigma^+<\sigma \tag{4.14}$$

Which means the position difference would decrease from period to period and eventually synchronized, and so does the phase difference.

Thus, the two robots are guaranteed to synchronize with each other eventually. With this result, we can state the conjecture that this sync process remains true for arbitrary large numbers of robots greater than 2, and the sync results of 50 and 100 robots has already been obtained from simulations as shown in Fig. 3.3 and Fig. 3.4.

Chapter 5

Conclusions and future work

5.1 Conclusion

In this thesis, we present a new type of swarm of decentralized robots that indirectly coupled with each other and can gradually synchronize with each other. The mechanical design is realized by 3D printing technology. With only one rear leg driven by a geared DC motor along with the other front leg to support itself, the robot can walk forward with oscillatory gaits. Then we design and implement the *Hopf adaptive controller* for our robot. This type of controller works as a "contact force detector" and provide the robot with "proprioceptive" ability to interact with the environment without adding any additional sensors. Both simulation and experiment platforms are properly set up and we generated a great deal of results from them. With the *mean resultant length* statistical concept, we are able to quantify the relationship among robots' phases, so that we can mathematically define the degree of sync behavior. Besides, some further explanations are given according to some typical phenomena occurs in the results. From the results, we find the rule of selecting parameters for our controller so we can improve synchronization. In addition, the effectiveness of sync compared with non-sync cases is verified, as the synchronized robots is more efficient and can push the obstacle with a lot more weight forward.

In the latter part of this thesis, we deeply analyse the rules behind synchronization, especially for two robots. First, some basic analyses with the return map from period to period are presented. Then, we further discover the detailed synchronization process in one period by the definition of three different contact patterns. Finally, based on all the results and analyses, we establish a simplified mathematical model and prove the synchronization result of two robots. Thus, we highly understand the property of our controller and the mechanism behind the synchronization.

5.2 Future work

There are still several aspects we may improve in the future work. Now our adaptive controller is quite simple and there are lots of other types of controllers we may apply to our robot (e.g. the Van der Pol oscillator equation). Besides, the proof we make for sync is derived from the simplified model, which ignored a lot dynamic process. A complete physical model should be established if we want to further study the mechanism behind sync precisely. Also In this thesis we mostly explain the interactions between two robots and the obstacle, a more general version of explanation should be provide for arbitrary numbers of robots.

Currently we only carry out the experiments for four robots, in the future, we can build more robots and verify the synchronization behavior with extremely large scale of robot swarms. In addition, the "pushing against obstacle" model in this thesis is simply a one dimension model, since the motions and force interactions only occur towards a single direction. If the robot can be redesigned to have the steering ability (e.g. use two rear legs to enable differential steering), then we may do some further experiments with more robots in a two dimension set up. A demonstration in simulation is shown in Fig. 5.1, we set up the robots in a square arena with different initial positions and orientations. We guide the robots to steer and walk towards the nearest obstacle, then we can observe the interactions among all the robots by pushing against

the obstacle or contacting with each other. This would be a more realistic application with our adaptive control method.

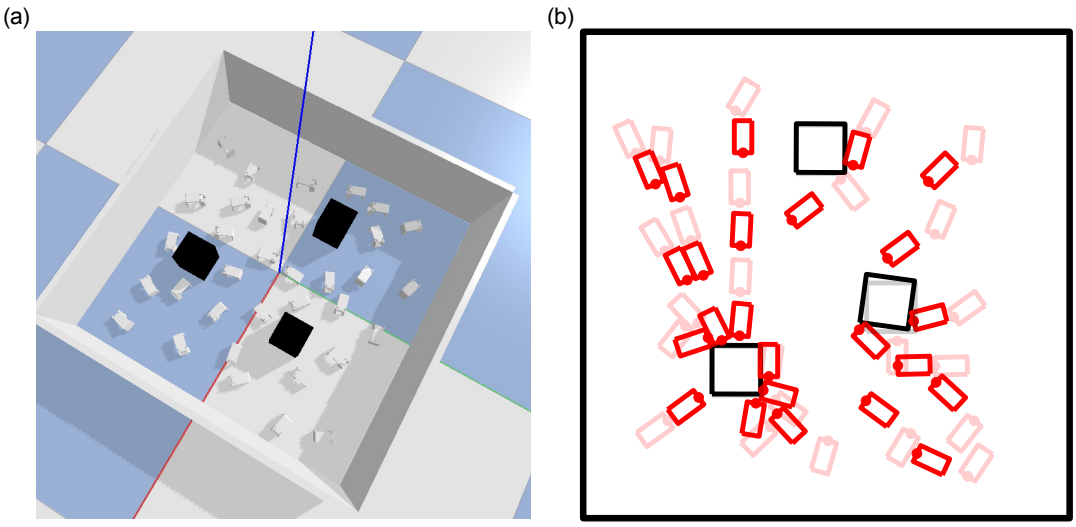

Figure 5.1: A simulation demonstration of robot swarms in an arena. (a) A screenshot from Pybullet. (b) The movement process in this arena simulation (initial positions are shown in transparent red, final positions are shown in solid red).

Bibliography

[1] Yoshiki Kuramoto. *Chemical oscillations, waves, and turbulence*. Courier Corporation, 2003.

[2] John Buck. Synchronous rhythmic flashing of fireflies. ii. *The Quarterly review of biology*, 63(3):265–289, 1988.

[3] Hugh M Smith. Synchronous flashing of fireflies. *Science*, 82(2120):151–152, 1935.

[4] Rothschild. Measurement of sperm activity before artificial insemination. *Nature*, 163(4140):358–359, 1949.

[5] Geoffrey Ingram Taylor. Analysis of the swimming of microscopic organisms. *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences*, 209(1099):447–461, 1951.

[6] Ingmar H Riedel, Karsten Kruse, and Jonathon Howard. A self-organized vortex array of hydrodynamically entrained sperm cells. *Science*, 309(5732):300–303, 2005.

[7] John Jalife. Mutual entrainment and electrical coupling as mechanisms for synchronous firing of rabbit sino-atrial pace-maker cells. *The Journal of physiology*, 356(1):221–243, 1984.

[8] Donald C Michaels, Edward P Matyas, and Jose Jalife. Mechanisms of sinoatrial pacemaker synchronization: a new hypothesis. *Circulation research*, 61(5):704–714, 1987.

[9] Charles S Peskin. *Mathematical aspects of heart physiology*. Courant Institute of Mathematical Sciences, New York University, 1975.

[10] Arthur Sherman and John Rinzel. Collective properties of insulin-secreting cells. In *Cell to Cell Signalling*, pages 61–75. Elsevier, 1989.

[11] Arthur Sherman, John Rinzel, and Joel Keizer. Emergence of organized bursting in clusters of pancreatic beta-cells by channel sharing. *Biophysical journal*, 54(3):411–425, 1988.

[12] James T Enright. Temporal precision in circadian systems: a reliable neuronal clock from unreliable components? *Science*, 209(4464):1542–1545, 1980.

[13] Theodosios Pavlidis. *Biological oscillators: their mathematical analysis*. Elsevier, 2012.

[14] Renato E Mirollo and Steven H Strogatz. Synchronization of pulse-coupled biological oscillators. *SIAM Journal on Applied Mathematics*, 50(6):1645–1662, 1990.

[15] Aviram Gelblum, Itai Pinkoviezky, Ehud Fonio, Abhijit Ghosh, Nir Gov, and Ofer Feinerman. Ant groups optimally amplify the effect of transiently informed individuals. *Nature communications*, 6(1):1–9, 2015.

[16] Christian Peeters and Stéphane De Greef. Predation on large millipedes and self-assembling chains in leptogenys ants from cambodia. *Insectes Sociaux*, 62(4):471–477, 2015.

[17] Sarah R Ho, Richard Smith, and Damien O'Meara. Biomechanical analysis of dragon boat paddling: a comparison of elite and sub-elite paddlers. *Journal of sports sciences*, 27(1):37–47, 2009.

[18] Arthur T Winfree. Biological rhythms and the behavior of populations of coupled oscillators. *Journal of theoretical biology*, 16(1):15–42, 1967.

[19] Steven H Strogatz. From kuramoto to crawford: exploring the onset of synchronization in populations of coupled oscillators. *Physica D: Nonlinear Phenomena*, 143(1-4):1–20, 2000.

[20] Jonas Buchli and Auke Jan Ijspeert. Distributed central pattern generator model for robotics application based on phase sensitivity analysis. In *International Workshop on Biologically Inspired Approaches to Advanced Information Technology*, pages 333–349. Springer, 2004.

[21] Eugene M Izhikevich and Yoshiki Kuramoto. Weakly coupled oscillators. *Encyclopedia of mathematical physics*, 5:448, 2006.

[22] Daniel Kaplan and Leon Glass. *Understanding nonlinear dynamics*. Springer Science & Business Media, 1997.

[23] Auke Jan Ijspeert. Central pattern generators for locomotion control in animals and robots: a review. *Neural networks*, 21(4):642–653, 2008.

[24] Auke Jan Ijspeert, Alessandro Crespi, Dimitri Ryczko, and Jean-Marie Cabelguen. From swimming to walking with a salamander robot driven by a spinal cord model. *science*, 315(5817):1416–1420, 2007.

[25] Erol Şahin. Swarm robotics: From sources of inspiration to domains of application. In *International workshop on swarm robotics*, pages 10–20. Springer, 2004.

[26] Shuguang Li, Richa Batra, David Brown, Hyun-Dong Chang, Nikhil Ranganathan, Chuck Hoberman, Daniela Rus, and Hod Lipson. Particle robotics based on statistical mechanics of loosely coupled components. *Nature*, 567(7748):361–365, 2019.

[27] David L Christensen, Srinivasan A Suresh, Katie Hahm, and Mark R Cutkosky. Let's all pull together: Principles for sharing large loads in microrobot teams. *IEEE Robotics and Automation Letters*, 1(2):1089–1096, 2016.

[28] Michael Rubenstein, Alejandro Cornejo, and Radhika Nagpal. Programmable self-assembly in a thousand-robot swarm. *Science*, 345(6198):795–799, 2014.

[29] Michael Rubenstein, Adrian Cabrera, Justin Werfel, Golnaz Habibi, James McLurkin, and Radhika Nagpal. Collective transport of complex objects by simple robots: theory and experiments. In *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*, pages 47–54, 2013.

[30] Erwin Coumans and Yunfei Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning. `http://pybullet.org`, 2016–2019.

[31] Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y Ng. Ros: an open-source robot operating system. In *ICRA workshop on open source software*, volume 3, page 5. Kobe, Japan, 2009.

[32] Nicholas I Fisher. *Statistical analysis of circular data*. cambridge university press, 1995.

[33] Philipp Berens. Circstat: a matlab toolbox for circular statistics. *Journal of Statistical Software, Articles*, 31(10):1–21, 2009.

[34] James Pantaleone. Synchronization of metronomes. *American Journal of Physics*, 70(10):992–1000, 2002.

[35] Steven H Strogatz, Daniel M Abrams, Allan McRobie, Bruno Eckhardt, and Edward Ott. Crowd synchrony on the millennium bridge. *Nature*, 438(7064):43–44, 2005.