Pulse-Coupled Oscillator Networks: Achieving Phase Continuity and Learning Optimal Control in Physical Systems

Timothy Anglea
tbangle@g.clemson.edu

Follow this and additional works at: https://tigerprints.clemson.edu/all_dissertations

Part of the Controls and Control Theory Commons

Recommended Citation
https://tigerprints.clemson.edu/all_dissertations/3063

This Dissertation is brought to you for free and open access by the Dissertations at TigerPrints. It has been accepted for inclusion in All Dissertations by an authorized administrator of TigerPrints. For more information, please contact kokeefe@clemson.edu.
Pulse-Coupled Oscillator Networks: Achieving Phase Continuity and Learning Optimal Control in Physical Systems

A Thesis
Presented to
the Graduate School of
Clemson University

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy
Electrical Engineering

by
Timothy Benjamin Anglea
May 2022

Accepted by:
Dr. Yongqiang Wang, Committee Chair
Dr. Richard Groff
Dr. Ian D. Walker
Dr. Umesh Vaidya
Abstract

In this dissertation, we consider the application of pulse-coupled oscillator theory to real-world, physical networks. When the phase of an oscillator is associated with a physical measure, such as clock timing or robotic heading, discontinuous adjustments of the oscillator's phase is undesirable and potentially disadvantageous. Rather, continuous adjustment of the oscillator phase value is needed over a certain amount of time. To ensure that both synchronization and desynchronization can still be achieved under the constraint of continuous phase value changes, we pursue a novel approach to analyze the generalization of a pulse-coupled oscillator network with a time-varying coupling strength. We provide rigorous mathematical proof for both pulse-coupled synchronization and desynchronization under the proposed phase continuity methods. We then correlate the continuous phase change of the oscillator to a specifically time-varying coupling strength of the network. To verify the analysis, we provide both simulated and experimental results for various synchronization and desynchronization algorithms using the proposed phase continuity methods.

Additionally, an oscillator may need to adjust its phase response to received pulses from connected neighbors in the network due to non-ideal conditions of physical systems, such as pulse propagation delay, non-identical oscillator frequencies, and general network topologies, once the network has been deployed. Direct analysis of pulse-coupled oscillator networks under non-ideal conditions is difficult, so we consider a novel approach of using reinforcement learning techniques to have the oscillators use their own experience to approximate an optimal phase response. Using appropriate measures to have a single oscillator estimate the state of the rest of the network, we determine a novel phase response function model in terms of the network topology. The optimality of the proposed phase response function is verified with simulated comparisons to existing synchronization algorithms.
This dissertation is dedicated to my late childhood friend Josiah (Joe) Paul Matthew, who passed away on April 12th, 2016. He was unable to complete his work toward a Ph.D. in Chemistry at the University at Buffalo in New York. While this work cannot replace what he might have accomplished had God allowed him to live in this world longer, his memory has helped me as I pursued my own advanced degree. - I Peter 4:19
Acknowledgments

I would first like to thank my advisor and professor at Clemson University, Dr. Yongqiang Wang, for introducing me to the concepts and applications of pulse-coupled oscillators. His help and guidance in my research endeavors has been invaluable.

I must also thank the many undergraduate students of the Robotic Networks Creative Inquiry over the last seven years, who have helped immensely in the development and testing of the various aspects of the Roomba robotic network platform, specifically Edward Bear, Madeline Corvin, Joshua Harvey, Martyn Lemon, and Weihang Tan. The experimental results of pulse-coupled oscillators on a physical system could not have been achieved without their assistance.

I would also like to thank Dr. Yuanzhao Zhang and Dr. Steven Strogatz from the Santa Fe Institute in Santa Fe, New Mexico. They provided extremely helpful discussions and improvements to portions of the work that make up this dissertation.

Additional thanks go to Dr. Richard Groff and Dr. Ian Walker of the Electrical & Computer Engineering Department and Dr. Umesh Vaidya of the Mechanical Engineering Department at Clemson University for their help in reviewing this dissertation and their willingness to aid me in my degree as committee members.

I would also like to give thanks to my parents, Steven Mark and Patricia Ruth Anglea, who have given to me both encouragement and support throughout my academic pursuits in higher education. They may not completely understand what it is that I do in my research, but they are still proud of what I’ve been able to accomplish in my time at Clemson University.

Finally, I would like to thank all others who reviewed the initial drafts of this dissertation and the works leading up to it. Their insights and corrections are greatly appreciated.
# Contents

Title Page ................................................................. i
Abstract ........................................................................ ii
Dedication ...................................................................... iii
Acknowledgments ............................................................ iv
List of Figures ................................................................. vii

1 Introduction ................................................................. 1
  1.1 The Pulse-coupled Oscillator Model .......................... 1
  1.2 Main Contributions ............................................... 3
  1.3 Summary .............................................................. 7

2 Pulse-Coupled Oscillators Preliminaries ........................ 8
  2.1 Oscillator Phase Evolution ....................................... 8
  2.2 Graph Formulations ............................................... 13
  2.3 Methods for Achieving Phase Continuity .................. 13

3 Clock Synchronization with Phase Continuity .................. 16
  3.1 Oscillator Analysis ................................................ 16
  3.2 Synchronization Analysis ....................................... 18
  3.3 Simulations and Experimental Results ...................... 23

4 Decentralized Heading Control under Rate Constraints .... 46
  4.1 Pulse-Based Heading Coordination .......................... 46
  4.2 Synchronization under Heading Rate Constraint .......... 48
  4.3 Desynchronization under Heading Rate Constraint ....... 50
  4.4 Experimental Results .......................................... 57

5 Optimal Control Using Reinforcement Learning ............... 66
  5.1 Preliminaries for Non-Ideal Factors ......................... 67
  5.2 Reinforcement Learning Setup ............................... 68
  5.3 Experimental Results .......................................... 77
  5.4 Comparison to Existing Results ............................. 86

6 Conclusion ................................................................. 100
  6.1 Additional Conclusions from Chapter 3 ..................... 101
  6.2 Additional Conclusions from Chapter 4 ..................... 101
  6.3 Additional Conclusions from Chapter 5 ..................... 102
Appendices .............................................. 103
  A  Proofs of Lemmas .................................... 104

Bibliography ............................................. 110
List of Figures

1.1 Illustration of disadvantages of discontinuous clock synchronization. (a) Synchronization occurs after a jump forward of two time units, and the scheduled process is not executed. (b) Synchronization occurs after a jump backward of two time units, and the scheduled process is executed twice. ................................................. 4

2.1 The delay-advance phase response curve (PRC) to achieve phase synchronization given in (2.2) .......................................................... 10

2.2 The delay-advance phase response curve (PRC) for phase synchronization given in (2.2) with an example refractory period \( D = 0.4c_\phi \). .......................... 11

2.3 The phase response function (PRF) to achieve phase desynchronization given in (2.6) for \( N = 5 \) oscillators with coupling strength constants \( l_1 = 0.8 \) and \( l_2 = 0.6 \). . . . . 12

3.1 Phase evolution of a PCO network using the PRC given in (2.2) for \( N = 6 \) oscillators in an all-to-all topology, with \( \alpha = 0.5 \), refractory period \( D = 0.001 \), and random initial conditions in a containing arc \( \Lambda < \frac{1}{2} \). (a) Continuous phase evolution under the constant frequency method, with \( \omega_a = 0.3\omega_0 \). (b) Continuous phase evolution under the constant time method, with \( \tau = 0.3 \) seconds. (c) Phase jumps, for comparison to the phase continuity methods. .................................................. 24

3.2 Containing arcs, \( \Lambda \), as a function of time for the networks in Fig. 3.1. The convergence speed of the containing arcs under the constant frequency method, with \( \omega_a = 0.3\omega_0 \), and the constant time method, with \( \tau = 0.3 \) seconds, is reduced compared with the phase jump case. .................................................. 25

3.3 Phase evolution of a PCO network using the PRC given in (2.2) for \( N = 6 \) oscillators in an all-to-all topology, with \( \alpha = 0.5 \), refractory period \( D = 0.5 \), and random initial conditions in a containing arc \( \Lambda < \frac{1}{2} \). (a) Continuous phase evolution under the constant frequency method, with \( \omega_a = 0.3\omega_0 \). (b) Continuous phase evolution under the constant time method, with \( \tau = 0.3 \) seconds. (c) Phase jumps, for comparison to the phase continuity methods. .................................................. 26

3.4 Containing arcs, \( \Lambda \), as a function of time for the networks in Fig. 3.3. The convergence speed of the containing arcs under the constant frequency method, with \( \omega_a = 0.3\omega_0 \), and the constant time method, with \( \tau = 0.3 \) seconds, is reduced compared with the phase jump case. .................................................. 27

3.5 Phase evolution of a PCO network using the PRC given in (2.2) for \( N = 6 \) oscillators in a ring topology, with \( \alpha = 0.5 \), refractory period \( D = 0.001 \), and random initial conditions in a containing arc \( \Lambda < \frac{1}{2} \). (a) Continuous phase evolution under the constant frequency method, with \( \omega_a = 0.3\omega_0 \). (b) Continuous phase evolution under the constant time method, with \( \tau = 0.3 \) seconds. (c) Phase jumps, for comparison to the phase continuity methods. .................................................. 28
3.6 Containing arcs, $\Lambda$, as a function of time for the networks in Fig. 3.5. The convergence speed of the containing arcs under the constant frequency method, with $\omega_a = 0.3\omega_0$, and the constant time method, with $\tau = 0.3$ seconds, is reduced compared with the phase jump case. ................................................................. 29

3.7 Phase evolution of a PCO network using the PRC given in (2.2) for $N = 6$ oscillators in a line topology, with $\alpha = 0.5$, refractory period $D = 0.001$, and random initial conditions in a containing arc $\Lambda < \frac{1}{2}$. (a) Continuous phase evolution under the constant frequency method, with $\omega_a = 0.3\omega_0$. (b) Continuous phase evolution under the constant time method, with $\tau = 0.3$ seconds. (c) Phase jumps, for comparison to the phase continuity methods. ................................................................. 30

3.8 Containing arcs, $\Lambda$, as a function of time for the networks in Fig. 3.7. The convergence speed of the containing arcs under the constant frequency method, with $\omega_a = 0.3\omega_0$, and the constant time method, with $\tau = 0.3$ seconds, is reduced compared with the phase jump case. ................................................................. 31

3.9 Containing arcs, $\Lambda$, as a function of time in a network that does not converge for $N = 6$ oscillators in an all-to-all topology. The coupling strength was $\alpha = 0.51$ and the refractory period was $D = 0.001$. The initial conditions were randomly selected, resulting in a containing arc of length $\Lambda > \frac{1}{2}$. Under the constant frequency method ($\omega_a = 0.3\omega_0$), the oscillators were unable to synchronize, but under the constant time method ($\tau = 0.3$ seconds), the oscillators achieved synchronization. ................................................................. 32

3.10 Phase Response Curve (PRC) equivalents for the Peskin algorithm and Mirollo-Strogatz synchronization algorithms, and the Reachback Firefly Algorithm (RFA). ................................................................. 33

3.11 Containing arcs, $\Lambda$, as a function of time for the networks under the Peskin synchronization model with parameters $\epsilon = 0.002, \gamma = 3$. The convergence property of the containing arcs under the constant frequency method, with $\omega_a = 0.3\omega_0$, and the constant time method, with $\tau = 0.1$ seconds, is maintained compared to the phase jump case. ................................................................. 34

3.12 Containing arcs, $\Lambda$, as a function of time for the networks under the Mirollo-Strogatz synchronization model with parameters $\epsilon = 0.002, b = 5$. The convergence property of the containing arcs under the constant frequency method, with $\omega_a = 0.3\omega_0$, and the constant time method, with $\tau = 0.1$ seconds, is maintained compared to the phase jump case. ................................................................. 35

3.13 Containing arcs, $\Lambda$, as a function of time for the networks under the Reachback Firefly Algorithm (RFA) with parameter $\epsilon = 0.002$. The convergence property of the containing arcs under the constant frequency method, with $\omega_a = 0.007\omega_0$, and the constant time method, with $\tau = 1.1$ seconds, is maintained compared to the phase jump case. The RFA algorithm further allows for phase continuity parameters to be in a broader interval than the Peskin algorithm. ................................................................. 36

3.14 Raspberry Pi 3 Model B and Xbee RF communication module. ................................................................. 37

3.15 Phase evolution of a physical PCO network using the PRC given in (2.2) for $N = 6$ oscillators in an all-to-all topology, with $\alpha = 0.5$, refractory period $D = 0.001$, and random initial conditions in a containing arc $\Lambda < \frac{1}{2}$. (a) Continuous phase evolution under the constant frequency method, with $\omega_a = 0.3\omega_0$. (b) Continuous phase evolution under the constant time method, with $\tau = 3$ seconds. (c) Phase jumps, for comparison to the phase continuity methods. ................................................................. 37

3.16 Containing arcs, $\Lambda$, as a function of time for the networks in Fig. 3.15. The convergence speed of the containing arcs under the constant frequency method, with $\omega_a = 0.3\omega_0$, and the constant time method, with $\tau = 3$ seconds, is reduced compared with the phase jump case. ................................................................. 38
3.17 Phase evolution of a physical PCO network using the PRC given in (2.2) for \( N = 6 \) oscillators in an all-to-all topology, with \( \alpha = 0.5 \), refractory period \( D = 0.5 \), and random initial conditions in a containing arc \( \Lambda < \frac{1}{2} \). (a) Continuous phase evolution under the constant frequency method, with \( \omega_a = 0.3\omega_0 \). (b) Continuous phase evolution under the constant time method, with \( \tau = 3 \) seconds. (c) Phase jumps, for comparison to the phase continuity methods.

3.18 Containing arcs, \( \Lambda \), as a function of time for the networks in Fig. 3.17. The convergence speed of the containing arcs under the constant frequency method, with \( \omega_a = 0.3\omega_0 \), and the constant time method, with \( \tau = 3 \) seconds, is reduced compared with the phase jump case.

3.19 Phase evolution of a physical PCO network using the PRC given in (2.2) for \( N = 6 \) oscillators in a ring topology, with \( \alpha = 0.5 \), refractory period \( D = 0.001 \), and random initial conditions in a containing arc \( \Lambda < \frac{1}{2} \). (a) Continuous phase evolution under the constant frequency method, with \( \omega_a = 0.3\omega_0 \). (b) Continuous phase evolution under the constant time method, with \( \tau = 0.3 \) seconds. (c) Phase jumps, for comparison to the phase continuity methods.

3.20 Containing arcs, \( \Lambda \), as a function of time for the networks in Fig. 3.19. The convergence speed of the containing arcs under the constant frequency method, with \( \omega_a = 0.3\omega_0 \), and the constant time method, with \( \tau = 0.3 \) seconds, is reduced compared with the phase jump case.

3.21 Phase evolution of a physical PCO network using the PRC given in (2.2) for \( N = 6 \) oscillators in a line topology, with \( \alpha = 0.5 \), refractory period \( D = 0.001 \), and random initial conditions in a containing arc \( \Lambda < \frac{1}{2} \). (a) Continuous phase evolution under the constant frequency method, with \( \omega_a = 0.3\omega_0 \). (b) Continuous phase evolution under the constant time method, with \( \tau = 0.3 \) seconds. (c) Phase jumps, for comparison to the phase continuity methods.

3.22 Containing arcs, \( \Lambda \), as a function of time for the networks in Fig. 3.21. The convergence speed of the containing arcs under the constant frequency method, with \( \omega_a = 0.3\omega_0 \), and the constant time method, with \( \tau = 0.3 \) seconds, is reduced compared with the phase jump case.

3.23 Containing arcs, \( \Lambda \), as a function of time for the networks under the Peskin synchronization model with parameters \( \epsilon = 0.002, \gamma = 3 \). The synchronization of the network under the constant frequency method, with \( \omega_a = 0.3\omega_0 \), and the constant time method, with \( \tau = 0.1 \) seconds, is achieved more slowly when compared to the phase jump case.

3.24 Containing arcs, \( \Lambda \), as a function of time for the networks under the Mirollo-Strogatz synchronization model with parameters \( \epsilon = 0.002, b = 5 \). The synchronization of the network under the constant frequency method, with \( \omega_a = 0.3\omega_0 \), and the constant time method, with \( \tau = 0.1 \) seconds, is achieved more slowly when compared to the phase jump case.

4.1 Example evolution of two robots’ phase and heading values in response to received pulses for a network with \( N = 6 \) robots in an all-to-all topology. The robot heading changes when the robot phase variable deviates from the fundamental frequency, \( \omega_0 \), following the heading control given in (4.1).

4.2 Phase response functions (PRFs) to achieve desynchronization under the heading rate constraint. \( (N = 5, l_1 = 0.8, l_2 = 0.6, \omega_0 = 2\pi, \omega_{\text{max}} = 0.5\omega_0, t_0 = 0.1 \text{ seconds}) \)

4.3 Phase update rule based on phase response functions (PRFs) in Fig. 4.2. \( (N = 5, l_1 = 0.8, l_2 = 0.6, \omega_0 = 2\pi, \omega_{\text{max}} = 0.5\omega_0, t_0 = 0.1 \text{ seconds}) \)
4.4 The iRobot® Create 2 (Roomba) robotic platform used for experiments. Six Roombas are used to perform the experiments. Commands are sent to a Roomba using a Raspberry Pi 3 Model B microcomputer. Pulses are sent between Raspberry Pi 3s using a XBee RF module. Data is retrieved from on-board Roomba sensors to determine heading and orientation.

4.5 Heading evolution of the robots under the heading synchronization algorithm for \( N = 6 \) robots in an all-to-all topology when the heading rate constraint is ignored, with \( \alpha = 0.5, \omega_0 = \frac{\pi}{9}, \text{ and } \omega_{\text{max}} = 0.3\omega_0 \). When the heading adjustment is assumed to be instantaneous, the robots are unable to converge their headings.

4.6 (a) Heading evolution of the robots under the heading synchronization algorithm for \( N = 6 \) robots in an all-to-all topology, with \( \alpha = 0.5, D = 0, \omega_0 = \frac{\pi}{9}, \text{ and } \omega_{\text{max}} = 0.3\omega_0 \). (b) Containing arc, \( \Lambda \), as a function of time for the robot headings in (a).

4.7 (a) Heading evolution of the robots under the heading synchronization algorithm for \( N = 6 \) robots in a bidirectional ring topology, with \( \alpha = 0.5, D = 0, \omega_0 = \frac{\pi}{9}, \text{ and } \omega_{\text{max}} = 0.3\omega_0 \). (b) Containing arc, \( \Lambda \), as a function of time for the robot headings in (a).

4.8 (a) Heading evolution of the robots under the heading synchronization algorithm for \( N = 6 \) robots in an all-to-all topology, with \( \alpha = 0.5, D = \pi, \omega_0 = \frac{\pi}{9}, \text{ and } \omega_{\text{max}} = 0.3\omega_0 \). (b) Containing arc, \( \Lambda \), as a function of time for the robot headings in (a).

4.9 (a) Heading evolution of the robots under the heading desynchronization algorithm for \( N = 6 \) robots in an all-to-all topology, with \( l_1 = 0.8, l_2 = 0.6, \omega_0 = \frac{\pi}{9}, \text{ and } \omega_{\text{max}} = 0.3\omega_0 \). (b) Desynchronization measure, \( P \), as a function of time for the robot headings in (a).

4.10 (a) Heading evolution of the robots under the heading desynchronization algorithm for \( N = 6 \) robots in an all-to-all topology, with \( l_1 = 0.8, l_2 = 0, \omega_0 = \frac{\pi}{9}, \text{ and } \omega_{\text{max}} = 0.3\omega_0 \). (b) Desynchronization measure, \( P \), as a function of time for the robot headings in (a).

5.1 Average optimal policy learned using a network of two oscillators with identical frequencies and zero delay. The dotted lines show the maximum variations in the learned policies. The learned PRF closely approximates the analytical optimal solution in [54].

5.2 Average optimal PRF learned using six oscillators in an all-to-all topology with identical frequencies and zero delay. The dotted lines show the maximum variations in the learned policies. Since the oscillators have an unrestricted distribution of initial phases, the learned PRF is significantly different from the analytical prediction in [54] and is better modeled using (5.11).

5.3 Average optimal policy learned using a network of six oscillators in a bidirectional ring topology with identical frequencies and zero propagation delay. The dotted lines show the maximum variations in the learned policies. While the learned phase response is similar to that found in [54], it is better approximated using our response model due to the consideration of global initial phase values.

5.4 Average optimal policy learned using a network of three oscillators in an all-to-all topology with identical frequencies and zero propagation delay. The dotted lines show the maximum variations in the learned policies. The learned phase response is similar to that found in Fig. 5.3, since the oscillators have the same indegree in both network topologies.
5.5 Average values of the learned function parameters in (5.11) using random networks of various sizes. Error bars show the minimum and maximum values learned for a given indegree. Oscillators with the same indegree have similar learned PRFs of the form given by (5.11). A best-fit trendline for the average values of the form given by (5.12) shows that the optimal PRF can be predicted solely based on the indegree of the node. ................................................................. 82

5.6 Average optimal policy learned using a network of six oscillators in an all-to-all topology with non-identical frequencies and zero propagation delay. The phase response function was determined with oscillator frequencies uniformly chosen from the range $[1.98\pi, 2.02\pi]$. The dotted lines show the maximum variations in the learned policies. The learned PRF is similar to that found under ideal conditions. ......................... 83

5.7 Average optimal policy learned using a network of six oscillators in an all-to-all topology with non-identical frequencies and zero propagation delay. The phase response function was determined with oscillator frequencies uniformly chosen from the range $[1.90\pi, 2.10\pi]$. The dotted lines show the maximum variations in the learned policies. The learned PRF is similar to that found under ideal conditions. ......................... 84

5.8 Average optimal policy learned using a network of six oscillators in an all-to-all topology with identical oscillator frequencies. The phase response function was determined with constant pulse propagation delay $\tau_{ij} = 0.1 \forall i, j$. The dotted lines show the maximum variations in the learned policies. The learned PRF is similar to that found under ideal conditions. ......................... 85

5.9 Average optimal policy learned using a network of six oscillators in an all-to-all topology with identical frequencies. The phase response function was determined with uniformly random pulse propagation delay $\tau_{ij} \in [0, 0.1]$ for each episode. The dotted lines show the maximum variations in the learned policies. The learned PRF is similar to that found under ideal conditions. ......................... 86

5.10 Average optimal policy learned using a network of six oscillators in an all-to-all topology with non-identical frequencies and random propagation delay. The phase response function was determined with oscillator frequencies uniformly chosen from the range $[1.90\pi, 2.10\pi]$ and uniformly random pulse propagation delay $\tau_{ij} \in [0, 0.1]$ for each episode. The dotted lines show the maximum variations in the learned policies. The learned PRF is similar to that found under ideal conditions. ......................... 87

5.11 The average of the steady-state value of the containing arc for a network of six oscillators in an all-to-all topology with non-identical oscillator frequencies in the range $[1.99\pi, 2.01\pi]$ and uniformly random pulse propagation delay in the range $\tau_{ij} \in [0.02, 0.04]$ for 2,000 runs. Our learned PRF achieves a greater amount of synchronization for small coupling strength values. ......................... 88

5.12 The probability of synchronizing for a network of six oscillators in an all-to-all topology with non-identical oscillator frequencies in the range $[1.99\pi, 2.01\pi]$ and uniformly random pulse propagation delay in the range $\tau_{ij} \in [0.02, 0.04]$ for 2,000 runs. Our learned PRF synchronizes more frequently than the other synchronization strategies at lower coupling strength values. ......................... 89

5.13 The average of the steady-state value of the containing arc for a network of six oscillators in an all-to-all topology with non-identical oscillator frequencies in the range $[1.9\pi, 2.1\pi]$ and uniformly random pulse propagation delay in the range $\tau_{ij} \in [0, 0.04]$ for 2,000 runs. ......................... 90

5.14 The probability of synchronizing for a network of six oscillators in an all-to-all topology with non-identical oscillator frequencies in the range $[1.9\pi, 2.1\pi]$ and uniformly random pulse propagation delay in the range $\tau_{ij} \in [0, 0.04]$ for 2,000 runs. The learned PRF is able to achieve synchronization more often than the other strategies, especially for small coupling strengths. ......................... 91
5.15 The average of the steady-state value of the containing arc for a network of eight oscillators in a ring topology with non-identical oscillator frequencies in the range $[1.98\pi, 2.02\pi]$ and uniformly random pulse propagation delay in the range $\tau_{ij} \in [0.01, 0.02]$ for 2,000 runs. Our learned PRF achieves a greater amount of synchronization over the broadest range of coupling strength values.

5.16 The probability of synchronizing for a network of eight oscillators in a ring topology with non-identical oscillator frequencies in the range $[1.98\pi, 2.02\pi]$ and uniformly random pulse propagation delay in the range $\tau_{ij} \in [0.01, 0.02]$ for 2,000 runs. Our learned PRF synchronizes more frequently than the Mirollo-Strogatz algorithm [26] at lower coupling strength values.

5.17 The oscillator network topology used for Fig. 5.18 and Fig. 5.19. Each oscillator node is connected to its two nearest neighbor on both sides.

5.18 The average of the steady-state value of the containing arc for a network of eleven oscillators in the “Near4” topology with non-identical oscillator frequencies in range $[1.98\pi, 2.02\pi]$ and uniformly random pulse propagation delay in the range $\tau_{ij} \in [0.01, 0.10]$ for 2,000 runs. Our learned PRF achieves the greatest amount of synchronization over the broadest range of coupling strength values.

5.19 The probability of synchronizing for a network of eleven oscillators in the “Near4” topology with non-identical oscillator frequencies in range $[1.98\pi, 2.02\pi]$ and uniformly random pulse propagation delay in the range $\tau_{ij} \in [0.01, 0.10]$ for 2,000 runs. Our learned PRF is able to synchronize the network more frequently than the other synchronization strategies as the coupling strength is decreased.

5.20 The Erdös–Rényi-Gilbert (ERG) random network topology used for Fig. 5.21 and Fig. 5.22. The probability that any two oscillators are connected was set to 0.3.

5.21 Average of the steady-state containing arc for ten oscillators in a (ERG) random topology with non-identical oscillator frequencies and random coupling delays. Each run started with randomly selected phases in $S^1$ with a containing arc greater than half of the cycle length.

5.22 Probability of synchronizing for ten oscillators in a (ERG) random topology with non-identical frequencies and random delay. Each run started with randomly selected phases in $S^1$ with a containing arc greater than half of the cycle length. The learned PRF is able to achieve synchronization more often than the other strategies, especially for smaller coupling strengths.
The study and analysis of pulse-coupled oscillators (PCOs) is a currently active field of engineering research. The basic concept of a pulse-coupled oscillator (PCO) network is simple. Each node, i.e., an oscillator, in the network has a continuously cycling phase variable that increases at a constant rate. When an oscillator’s phase variable reaches a threshold value, the oscillator resets its phase to zero and sends, or fires, a single, simple pulse signal to all connected nodes in the network. The oscillator then continues its normal phase value evolution. When an oscillator receives a pulse signal, which is identical for each oscillator, it then modifies its phase variable according to some pre-determined phase adjustment algorithm. After this phase adjustment, the oscillator continues its normal phase value evolution until either another pulse is received or it reaches the threshold value and fires its own pulse.

1.1 The Pulse-coupled Oscillator Model

The PCO model was first introduced by Peskin in 1975, who used PCOs to model the synchronization of pacemaker cells in the heart [42]. PCOs, as well as appropriately designed phase adjustment algorithms, are also used to model the behavior of many other biological systems, such as the firing of neurons [42, 13], the flashing of fireflies [26, 20], rhythmic audience applause [32] and schools of fish [40].

A few years after Peskin’s introduction of the PCO model, Mirollo and Strogatz improved the model, providing a more rigorous mathematical formulation [26]. Up to the present time, the
PCO model has been used and developed by many others, who have presented ways to improve the model’s capabilities and characteristics, and extend its applications [See [12] and references within]. The primary benefits of the model are found through the minimization of communication latency, packet loss, signal corruption, and energy consumption, and are due to the simplicity of the communication between nodes in the network, i.e., single pulses [49]. Some practical applications of PCO networks include clock synchronization [57, 21, 55, 56] and round-robin scheduling and communication [41, 9, 39] in decentralized wireless sensor networks, along with planar robotic vehicle coordination [47, 48, 16].

There are two main behaviors – synchronization and desynchronization – that can be achieved in a PCO network. These behaviors are discussed in the following sections.

1.1.1 PCO Synchronization

The first main behavior of PCO networks is synchronization, where the oscillators in the network attempt to converge to an identical phase value through the resulting pulse-coupled interaction between the nodes.

Two types of oscillator synchronization are considered in the literature. The first type is called “strong” synchronization, where the oscillator phases converge and maintain the same value during the evolution of the network. The second is called “weak” synchronization, where only the pulse firings between the oscillators are aligned. In this second type of synchronization, the oscillator phases don’t necessarily have to be identical, but the phase adjustment algorithm is such that the oscillators always fire in unison. Strong synchronization of an oscillator network implies weak synchronization. For this dissertation, we will only consider strong synchronization.

Since the introduction of the model by Peskin [42] and the mathematical formulation provided by Mirollo and Strogatz [26], much research has been done analyzing the dynamics and behavior of PCO networks achieving synchronization. Optimal synchronization algorithms have been found in terms of time to convergence [54] as well as energy efficiency [55]. Under appropriate conditions, algorithm parameters, and network topologies, strong synchronization can be achieved from any initial condition of the oscillators [36, 35]. Additional research has been done to consider the effects of pulse propagation delay [34, 33, 24], oscillators with non-identical frequencies [5], minimally connected network topologies [44], and dynamic network topologies [23].
1.1.2 PCO Desynchronization

The second main behavior of PCO networks is desynchronization, the inverse of synchronization, where the oscillators in the network attempt to diverge the phase values between all nodes and have the phase values spread apart as much as possible.

Like with synchronization, there are two types of oscillator desynchronization considered in the literature. The first type is called “strong” desynchronization, where the oscillator phases are spread out evenly from each other. The second type is called “weak” desynchronization, in which the oscillators fire pulses at even, equal, intervals [41, 43]. In this type of desynchronization, oscillator phases may not necessarily be evenly spaced from each other, and may be adjusting dynamically to maintain the even spacing of the pulse firings. Strong desynchronization of an oscillator network implies weak desynchronization. For this dissertation, we will only consider strong desynchronization.

Much less research has been done with the analysis of desynchronizing PCO networks than with synchronizing. Given an appropriate choice of desynchronization algorithm, asymptotic stability of both weak desynchronization [43] and strong desynchronization [14] can be achieved. Work has also been done to improve the speed of convergence assuming an all-to-all topological network [53, 7].

Nearly all existing results on the desynchronization of PCO networks requires the condition of an all-to-all connected network topology and a fixed number of oscillators [41, 9, 39, 10, 15]. Relaxing these condition necessitates additional limitations to other aspects of the network, such as the initial phase of the oscillators [2], or instead require the use of a continuously coupled, rather than pulse-coupled, oscillators [47, 48], or the use of additional types of nodes and pulse transmissions [17].

1.2 Main Contributions

1.2.1 Clock Synchronization

Ensuring clock synchronization in a distributed system is a very important and well-studied topic in the fields of electrical engineering and computer science. However, many synchronization algorithms require an instantaneous clock value adjustment [46], which results in clocks with dis-
continuous time stamps. These discontinuous clocks are undesirable, especially if certain events or processes are time-dependent. If time jumps forward, then there is the potential that a scheduled event will never happen, and if time jumps backwards, then one process may be implemented twice, as shown in Fig. 1.1. It is desirable that a clock’s time value evolves continuously while synchronizing with other clocks within the network [25, 50].

Many clock synchronization algorithms use packet-based communication to share local information [28, 45, 51] and can achieve synchronization with a continuous clock [25, 50]. However, as noted in [45], such approaches may require constant adjustment to the clock rate, leading to significant runtime overhead. Recently, the PCO model has been widely used to synchronize clocks
in wireless sensor networks.

1.2.2 Robotic Heading Control

Driven by recent accelerated technological advances, mobile robot networks are receiving increased attention, being widely used in warehouse management, surveillance, reconnaissance, search and rescue, and even cooperative construction. Efficient and effective coordination of robotic networks requires advanced cooperative control algorithms, which have been extensively studied in the past decade [6, 38, 31].

However, due to the disparate discrete-time nature of communication and the continuous-time evolution of motion dynamics, robotic network coordination intrinsically inherits hybrid dynamics, which makes its rigorous treatment very difficult. In fact, to simplify the design, many existing results (e.g., [40, 47, 48, 8, 30]) choose to design the decentralized controller in the continuous-time domain and then discretize the controller in implementation to conform to the discrete-time nature of communication. Such a design makes the obtained coordination algorithm sensitive to discretization errors, negatively affecting the achievable coordination accuracy.

Another difficulty in motion coordination lies in the control input constraint due to a finite speed of heading adjustment, which adds more nonlinearity to the already nonlinear robot network coordination problem. Note that in the coordination of robot headings, since robot heading evolves in the nonlinear one-dimension torus, existing approaches addressing control-constrained consensus on states that evolve in Euclidean space, such as [1, 58, 60, 27, 59, 11], are not applicable.

In this dissertation, we will consider an approach to achieve heading control in a decentralized network with a constraint on the rate of heading change using hybrid dynamics inspired by oscillator networks. Recent research on complex oscillator networks, such as [18, 29], has studied their collective behavior and the ability to control populations of oscillators. However, those results require continuous coupling between oscillators, and do not lend themselves well to the discrete nature of communication in robotic networks. Thus, we will focus on the pulse-coupled oscillator model. Not only is the approach completely decentralized, but it also avoids the discretization problem, as the hybrid dynamics take into account the continuous evolution of motion and the discrete nature of communication explicitly.

We apply pulse-coupled oscillator strategies to control the heading, $\theta$, of a group of mobile robots using the analogy between oscillator phase and robot headings which both evolve in the one-
dimensional torus. More specifically, by relating the robot heading to the phase of an oscillator, we can coordinate the robot headings using phase coordination mechanisms based on the pulse-coupled oscillator model. With the many beneficial properties of PCO networks, PCO synchronization and desynchronization strategies seem ideal for the decentralized control of a swarm of mobile robots [16].

However, all of the existing PCO synchronization and desynchronization algorithms achieve convergence through abrupt jumps in the oscillator phase, requiring instantaneous change to the phase variable. As indicated earlier, these sudden changes may lead to undesired behavior when the phase variable is representative of network time, and direct application of these PCO strategies is not feasible in actual heading control. In this dissertation, we propose a generalization of the standard PCO model, which does not require instantaneous change in phase to achieve oscillator phase coordination and can guarantee continuity in phase evolution.

We will show that the convergence properties of previously proposed jump-based synchronization and desynchronization algorithms in [55, 54, 15] are maintained, even when the phase adjustment rule is modified to be continuous. To do so, we will prove that the generalization that guarantees phase continuity amounts to a reduction of the coupling strength in the standard PCO model. To our knowledge, this work is the first to analyze a PCO network with a time-varying coupling strength.

The generalized PCO model can also achieve proper phase coordination under an arbitrary constraint on the rate of phase adjustment. To reconcile the constantly evolving phase and the communication-triggered heading update, we also propose a heading coordination mechanism to control robot headings in the presence of heading rate constraints. The results address the hybrid dynamics of continuous-time motion evolution and discrete-time communication directly and hence can be applied in implementation without discretization.

1.2.3 Optimal Phase Update

Because ensuring phase continuity in a physical network can be modeled as a potential reduction in the coupling strength parameter, previously proposed synchronization strategies may not be able to guarantee that the network will synchronize. The addition of practical, non-ideal environmental factors such as non-identical oscillator frequencies and non-zero pulse propagation delays complicate the analysis of determining an optimal phase response function that allows the
network to synchronize from any initial condition.

Rather than directly analyzing the oscillator network, we will utilize reinforcement learning strategies to have the network itself determine an optimal response to incoming pulses from neighboring oscillators. This novel approach to phase response function design allows us to determine a synchronization strategy that maximizes the network’s ability to quickly and frequently synchronize, regardless of network topology or non-ideal environmental factors. We’ll discuss our reinforcement learning approach and compare our learned phase response function with proposed synchronization strategies found using analytical techniques. We find that our learned phase response function allows arbitrary oscillator network topologies to synchronize more often, especially in non-ideal environments, as the coupling strength is decreased.

1.3 Summary

In Chapter 2, we will formalize the definitions and notation regarding pulse-coupled oscillator networks and the proposed phase continuity algorithms. After that, we will follow in Chapter 3 with our rigorous proof for achieving clock synchronization while guaranteeing phase continuity using pulse-coupled oscillators. In Chapter 4, we will expand on our previous results to achieve synchronization and desynchronization of heading under phase continuity as a rate constraint. Then, we will explore our reinforcement learning approach to optimize and find a phase response function that synchronizes an arbitrary pulse-coupled oscillator network subject to practical non-ideal environmental factors in Chapter 5. Verification of our results will be provided through physical experiments and computer simulations in each corresponding chapter. Finally, we will offer summary conclusions in Chapter 6.
Chapter 2

Pulse-Coupled Oscillators

Preliminaries

In this chapter, we will discuss the various preliminary definitions and notations that we will use throughout the dissertation. In 2.1, we will discuss our notation for PCOs and our methods and measures of achieving synchronization and desynchronization in PCO networks. We’ll then detail some notation of PCOs from the perspective of graph theory in 2.2. We’ll conclude in 2.3 with a discussion of methods to achieve phase continuity in PCO networks.

2.1 Oscillator Phase Evolution

Consider a network of $N$ identical pulse-coupled oscillators. Let $\phi_i \in \mathbb{S}^1 = [0, c_\phi)$ be the associated phase of oscillator $i \in \mathcal{V} = \{1, 2, \cdots, N\}$, where $\mathbb{S}^1$ is the one-dimensional torus and $c_\phi$ is the phase threshold value for the oscillators. In [55, 14, 15, 2], $c_\phi$ is chosen to be $2\pi$, but is chosen to be 1 in [41, 39]. We will use the threshold value $c_\phi = 1$ in Chapter 3 and the threshold value $c_\phi = 2\pi$ in Chapter 4 and Chapter 5. Each oscillator evolves its phase at a rate $\omega_i$. For our analysis, unless specified otherwise, each oscillator has an identical fundamental frequency, $\omega_0$ (i.e., $\omega_1 = \omega_2 = \cdots = \omega_N = \omega_0$), and evolves naturally at that rate on the interval $[0, c_\phi)$.

During the normal evolution of the network, when an oscillator reaches the threshold value, it fires a pulse and resets its phase to zero. Any connected oscillators then receive that pulse, being
notified of the firing instance of an oscillator in the network. Receiving a pulse will cause an oscillator to change its phase in accordance with the chosen PCO algorithm. Let us denote the amount that oscillator \( i \) determines to adjust its phase at a firing instance at time \( t \) as

\[
\psi_i = F(\phi_i) = \lim_{\tau \downarrow 0} (\phi_i(t + \tau) - \phi_i(t)) = \phi_i(t^+) - \phi_i(t^-)
\]  

(2.1)

In (2.1), \( \psi_i \) represents the amount of phase change determined by the PCO algorithm, and \( \phi_i(t^-) \) and \( \phi_i(t^+) \) represent the phase of oscillator \( i \) before and after receiving a pulse, respectively.

The function \( F(\phi_i) \) is called a phase response function (PRF), or phase response curve (PRC), and is used to determine the amount that an oscillator will adjust its phase as a function of the oscillator’s phase when a pulse is received [22]. By an appropriate choice for the PRC and oscillator network topology, the phases of the oscillators in the network can converge to a state of phase synchronization or a state of phase desynchronization.

2.1.1 Synchronization Condition

A variety of algorithms and phase response functions have been proposed to ensure synchronization of oscillator phases in a PCO network. Many synchronization algorithms, including those in [55, 54, 36, 34] use a delay-advance PRC to describe the phase update at firing instances. Here, we define a phase response curve \( F(\phi) \) as a delay-advance phase response curve if \( F(\phi) \) causes the oscillator to adjust its phase backward during the first half of the oscillator cycle and to adjust its phase forward during the last half of the cycle. Expressed mathematically, a delay-advance PRC satisfies the following conditions: 

\[
0 > F(\phi) \geq -\phi \text{ for } \phi \in (0, \frac{1}{2}c_\phi), \quad 0 < F(\phi) \leq c_\phi - \phi \text{ for } \phi \in (\frac{1}{2}c_\phi, c_\phi),
\]

\[-\phi \leq F(\phi) \leq c_\phi - \phi \text{ for } \phi = \frac{1}{2}c_\phi, \text{ and } F(\phi) = 0 \text{ for } \phi \in \{0, c_\phi\}.
\]

In our analysis of oscillator synchronization, we will focus on the PRC proposed in [55]. When an oscillator receives a pulse, it updates its phase variable according to the delay-advance phase response curve, or function, \( Q \), as shown in Fig. 2.1.

\[
Q(\phi_i) = \begin{cases} 
-\phi_i & \text{if } 0 \leq \phi_i \leq \frac{1}{2}c_\phi \\
(c_\phi - \phi_i) & \text{if } \frac{1}{2}c_\phi < \phi_i \leq c_\phi 
\end{cases}
\]  

(2.2)

Note that the phase update is independent of the number and relative positions of other
oscillators in the network. Thus, the phase of oscillator \( i \) after a firing instance can be described as

\[
\phi_i(t^+) = \phi_i(t) + \alpha Q(\phi_i(t))
\]  
(2.3)

where \( \alpha \) is the coupling strength of the network, and, from (2.1), we have \( \alpha Q(\phi_i(t)) = F(\phi_i) \).

Typical values for the range of the coupling strength \( \alpha \) are within the interval \((0, 1]\).

A refractory period of non-zero duration \( D \) can be included in the phase response curve, as shown in Fig. 2.2. An oscillator does not respond to incoming pulses if its phase is within the interval \([0, D)\), and continues to freely evolve. Such a refractory period can improve energy-efficiency and robustness to communication latency [55].

To evaluate how much a PCO network has synchronized, let us define an arc as a connected subset of the interval \( S^1 = [0, c_\phi) \). We can thus define the following set of functions, \( v_i \), for all \( i \in V \):

\[
v_i(\phi) = \min_{j \neq i} \{(\phi_j - \phi_i) \mod c_\phi\}
\]  
(2.4)
The function $v_i(\phi)$ represents the length of the arc along $S^1$ between oscillator $i$ and the first oscillator ahead of oscillator $i$. Note that $\sum_{i \in V} v_i(\phi) = c_\phi$ holds.

We then define the containing arc of the oscillators to be the smallest arc that contains all of the phases in the network. The length of this arc, $\Lambda$, is given mathematically as

$$
\Lambda \triangleq c_\phi - \max_{i \in V} \{v_i(\phi)\}
$$

(2.5)

When the oscillators phases converge to the same value, the length of the containing arc, $\Lambda$, approaches zero. Thus, the containing arc of the network gives us an appropriate measure to determine how well the oscillator phases have synchronized.

### 2.1.2 Desynchronization Condition

As with synchronization, a variety of algorithms and phase response functions have been proposed to achieve desynchronization of oscillators in a PCO network.

In our analysis of oscillator desynchronization, we will focus on the PRF proposed in [15].
Figure 2.3: The phase response function (PRF) to achieve phase desynchronization given in (2.6) for \(N = 5\) oscillators with coupling strength constants \(l_1 = 0.8\) and \(l_2 = 0.6\).

As shown in [15], the phase response function given by (2.6), where \(l_1 \in [0, 1)\) and \(l_2 \in [0, 1)\) are constants denoting the coupling strength such that \(l_1\) and \(l_2\) are not both zero simultaneously, can guarantee phase desynchronization in an all-to-all connection topology.

\[
F(\phi_i) = \begin{cases} 
  l_1 \left( \frac{c\phi}{N} - \phi_i \right) & 0 < \phi_i < \frac{c\phi}{N} \\
  0 & \frac{c\phi}{N} \leq \phi_i \leq \frac{c\phi N-1}{N} \\
  l_2 \left( \frac{c\phi N-1}{N} - \phi_i \right) & \frac{c\phi N-1}{N} < \phi_i < c\phi 
\end{cases} 
\]  

(2.6)

The function given by (2.6) is illustrated in Fig. 2.3 for a network of \(N = 5\) oscillators.

To quantify the degree of achievement toward phase desynchronization, we can use the measure given in [15]. From (2.4), we have the length of the arc between each oscillator and the oscillator directly ahead of it in phase denoted as \(v_i(\phi)\). Phase desynchronization implies that the arc length between neighboring oscillators is equal to \(\frac{c\phi}{N}\). Thus, the measure \(P\) given by (2.7) is used to quantify phase desynchronization based on these arc lengths.

\[
P \triangleq \sum_{i=1}^{N} \left| v_i(\phi) - \frac{c\phi}{N} \right| 
\]  

(2.7)
It is clear that this measure, \( P \), has a minimum of zero, and equals zero only when the oscillator phases are evenly distributed in the interval \( S^1 = [0,c\phi) \). Thus, this measure gives us an appropriate way to determine how well the oscillator phases have desynchronized.

### 2.2 Graph Formulations

To aid us in our analysis, a PCO network can be described as a directed graph \( G = (V,E) \), where \( V = \{v_1,v_2,\ldots,v_N\} \) is a set of vertices corresponding to the oscillators in the network, and \( E \) is the set of directed edges \((v_i,v_j)\) corresponding to the interaction between two oscillators [19]. An edge \((v_i,v_j)\) indicates that oscillator \( v_j \) can receive pulses from oscillator \( v_i \). A directed path is a sequence of edges \((v_i,v_j),(v_j,v_k),(v_k,v_\ell),\ldots\) in a graph \( G \). A graph \( G \) is considered to be strongly connected if there is a directed path between any pair of vertices in the graph.

For a vertex \( v_i \), the outdegree of the vertex, denoted \( \delta^+(v_i) \), is the number of edges that leave vertex \( v_i \). Similarly, the indegree of the vertex, denoted \( \delta^-(v_i) \), is the number of edges that enter vertex \( v_i \). Alternatively, the outdegree and indegree of a vertex is the number of edges in graph \( G \) that have that vertex as the first and second entry, respectively. The value \( \delta^+(G) \triangleq \min_{v \in V} \delta^+(v) \) is called the outdegree of the graph \( G \), and the value \( \delta^-(G) \triangleq \min_{v \in V} \delta^-(v) \) is called the indegree of the graph \( G \).

### 2.3 Methods for Achieving Phase Continuity

To our knowledge, all existing literature regarding PCO networks has the phase value of each oscillator jump discontinuously at firing instances. Here, we will generalize the standard PCO model such that the phase value must evolve continuously at all times (except when it resets its phase to zero when it reaches the threshold). To ensure this phase continuity, when oscillator \( i \) receives a pulse, it must increase or decrease its individual rate of evolution, \( \omega_i \), by a non-zero amount, \( \omega_a \), for a certain finite amount of time, \( \tau_i \), in order to achieve the required phase adjustment, \( \psi_i \), as in (2.1). If oscillator \( i \) receives another pulse before the time needed to achieve \( \psi_i \) is completed, then the oscillator will use its current phase \( \phi_i \) to redetermine a new \( \psi_i \), and thus determine new values for \( \omega_i \) and \( \tau_i \).

There are two primary methods for adjusting the phase of an oscillator in a continuous
fashion. In the first method, the oscillator, in response to a received pulse, achieves the desired phase adjustment by increasing or decreasing its frequency by a fixed amount. In the second method, the oscillator achieves the desired phase adjustment in a fixed amount of time.

2.3.1 Constant Frequency Method

In the constant frequency method, the frequency of oscillator \(i\) is increased or decreased by a set amount \(\omega_a\) for an adjustable duration of time \(\tau_i\). The amount of time the oscillator spends at this new frequency is dependent on the phase amount \(\psi_i\) that it needs to adjust:

\[
\tau_i = \frac{|\psi_i|}{\omega_a}
\]

Thus, once an amount of phase adjustment \(\psi_i\) is determined, the oscillator will increase its frequency to \(\omega_i = \omega_0 + \omega_a\) if \(\psi_i\) is positive, or decrease its frequency to \(\omega_i = \omega_0 - \omega_a\) if \(\psi_i\) is negative, for time \(\tau_i\) determined in (2.8). Once the time \(\tau_i\) has elapsed, the oscillator returns to its fundamental frequency \(\omega_0\). If \(\psi_i\) is zero, then the oscillator remains at its fundamental frequency, \(\omega_0\), and evolves until the next firing instance.

2.3.2 Constant Time Method

In the constant time method, oscillator \(i\) spends a fixed amount of time \(\tau\) at an adjustable frequency \(\omega_i\). The new frequency at which the oscillator evolves is dependent on the phase amount \(\psi_i\) the oscillator needs to adjust:

\[
\omega_i = \omega_0 + \omega_a = \omega_0 + \frac{\psi_i}{\tau}
\]

Note that \(\omega_a\) can be positive or negative. Thus, once an amount of phase adjustment \(\psi_i\) is determined, the oscillator will update its frequency to \(\omega_i = \omega_0 + \omega_a\) for time \(\tau\). Once the fixed amount of time \(\tau\) has elapsed, the oscillator again returns to its fundamental frequency \(\omega_0\).

2.3.3 Remarks on Methods

In the constant frequency method, it is possible to have different amounts of frequency change when increasing or decreasing the oscillator’s frequency, i.e., \(\omega_a^+\) and \(\omega_a^-\) respectively. Here,
we will focus on using the same amount of frequency change \( \omega_a = \omega_a^+ = \omega_a^- \) for both increasing and decreasing the frequency of the oscillator.

The constant time method described above is a general case of how other algorithms ensure clock continuity. In packet-based synchronization algorithms, the value of \( \tau \) is set to be the length of the communication period [50, 28].

Phase jumps, as is standard in the literature, can be seen as a specific case of either of the above phase continuity methods. These cases can be obtained by either taking the limit as \( \omega_a \) goes to infinity in the constant frequency method, or by taking the limit as \( \tau \) goes to zero in the constant time method.

Depending on the parameters chosen in each of the above phase continuity methods, the oscillators may evolve backward in phase (i.e., \( \omega_i < 0 \)). Negative frequencies are acceptable in the analysis used in this work. However, parameters can be chosen to ensure that \( \omega_i^- < 1 \) holds, such that the oscillator frequency, \( \omega_i \), remains strictly positive.

Both methods achieve the same basic result of having the phase evolve continuously. However, each has their desirable characteristics. The constant frequency method only requires the oscillators to evolve at a countable set of frequencies, and the effective coupling strengths for oscillators is maximized as the network approaches synchronization. The constant time method, however, ensures that the phase adjustment occurs in a set amount of time, resulting in more gradual changes in phase as the network synchronizes. The method used should take into consideration the specific application of the PCO network.
Chapter 3

Clock Synchronization with Phase Continuity

In Section 3.1, we will analyze the behavior of an oscillator under continuous phase evolution in a PCO network, and show that the behavior can be modeled as a bounded, time-varying coupling strength. In Section 3.2, we will analyze the convergence properties of PCOs under time-varying coupling strengths, and show that a PCO network will synchronize under guaranteed phase continuity. We will use numerical and hardware experimental results to evaluate the proposed approaches in Section 3.3. Throughout this chapter, we will use the phase threshold value \( c_\phi = 1 \).

3.1 Oscillator Analysis

We now rigorously analyze the dynamics of oscillators maintaining continuous phase evolution, rather than using phase jumps\(^1\). All that is required is to take the amount of phase adjustment for oscillator \( i \), i.e. \( \psi_i \), and determine the necessary amount of time, \( \tau_i \), and change in frequency \( \omega_i \). We then let the oscillator evolve at the new frequency for the required amount of time before it returns to its fundamental frequency, \( \omega_0 \).

\(^1\)This analysis has been published in the IEEE Transactions on Signal Processing, Vol. 67, No. 6. [3].
3.1.1 Single Oscillator Behavior

Let us analyze the behavior of a single oscillator. Once oscillator $i$ has received a pulse and calculated the necessary change in frequency, $\omega_i$, to achieve the phase adjustment $\psi_i$ in time $\tau_i$, two possibilities can follow: 1) the oscillator receives no new pulses within time $\tau_i$ that cause a phase adjustment, and 2) the oscillator receives a new pulse (i.e., the current pulse) within time $\tau_i$ that causes a phase adjustment. If the duration of the time interval between the current and previous oscillator firing instances is given as $t_0$, we can divide these two cases mathematically as 1) $t_0 \geq \tau_i$, and 2) $t_0 < \tau_i$.

1. In the first case, oscillator $i$ finishes adjusting its phase by $\psi_i$, and returns to evolving at the fundamental frequency $\omega_0$. The same effective change in phase has been achieved as if the oscillator had jumped in phase by $\psi_i$ and evolved normally for a time of length $t_0$. Thus, no effective change in the phase update rule occurs compared with the standard instantaneous jump-based PCO model.

2. In the second case, the oscillator has not yet achieved its desired amount of phase change. Rather than having adjusted the whole amount $\psi_i$, it has adjusted only a portion of that amount, $\frac{t_0}{\tau_i} \psi_i$, in the time interval between received pulses. The oscillator then will use its current phase at the time when the new pulse is received to redetermine new values for $\psi_i$, $\tau_i$, and $\omega_i$. This truncated phase evolution is equivalent to having the oscillator jump in phase by $\frac{t_0}{\tau_i} \psi_i$, and then evolve normally for a time of duration $t_0$. This fractional amount of the desired phase change can be viewed as a reduction of the coupling strength, $\alpha$, of oscillator $i$ by the ratio $\frac{t_0}{\tau_i}$.

From (2.1), both of these cases allow us to write a generalized expression for the effective coupling strength, $\alpha_{e_i}$, of the oscillator:

$$\alpha_{e_i} = \min\left\{\frac{t_0}{\tau_i} \alpha, \alpha\right\}$$

(3.1)

The quantity $\frac{t_0}{\tau_i}$ is greater than or equal to 1 in the first case, and less than 1 in the second case. Note that this expression is bounded by $(0, \alpha]$ and the value for an individual oscillator may vary over time as the PCO network evolves. This idea of an effective coupling strength leads us to analyze a standard PCO network under the condition of a time-varying coupling strength.
3.1.2 Time-Varying Coupling Strength

Let us consider a PCO network under a synchronization algorithm that allows jumps in the phase variable, $\phi$, but has a coupling strength, $\alpha$, that varies with time.

**Proposition 1.** The evolution of an oscillator in a PCO network is dependent upon the value of the coupling strength, $\alpha$, only at firing instances.

**Proof.** The proof for this proposition is straightforward. An oscillator only determines the amount that it needs to jump when it receives a pulse. Thus, the value of the coupling strength is only used at firing instances. Any values the coupling strength takes between firing instances is unused and thus independent of the behavior of the network. Furthermore, if the oscillator receives a pulse, but does not jump (or jumps an amount of zero), then the coupling strength is again independent to the behavior of the network.

**Remark 1.** The phase continuity modification described in Chapter 2 does not actually adjust the coupling strength as the network evolves. Only the apparent behavior of an oscillator is being modeled as a reduced coupling strength, $\alpha_e$. The actual coupling strength, $\alpha$, remains unchanged throughout the entire evolution of the network.

3.2 Synchronization Analysis

We will now use the PCO synchronization strategy given in [55] with phase jumps to determine if the convergence properties of the algorithm still hold under a time-varying coupling strength, and thus under the newly proposed phase continuity generalization.

3.2.1 Synchronization Condition

Consider a PCO network with $N$ oscillators in a (strongly) connected graph. We will use the delay-advance phase response curve, $Q$, given by (2.2), where we use $c_\phi = 1$, with a non-zero refractory period $D$. When an oscillator receives a pulse, it updates its phase variable according to $Q$ as shown in Fig. 2.2.

**Remark 2.** Due to the non-zero refractory period in a PRC, an oscillator does not respond to its own pulse firing. In fact, when oscillator $i$ fires, its phase is reset to zero, which is in the interval
and thus it will not respond to its own pulse, and will continue to evolve at its frequency \( \omega_i \),
determined by its response to the last pulse it received.

As proven in [55], the containing arc, \( \Lambda \), given in (2.5) will decrease and converge to zero
(i.e., the network will synchronize) under a constant coupling strength. Here, we show that \( \Lambda \) will
decrease even when the coupling strength is time-varying.

**Theorem 1.** Consider pulse-coupled oscillators with a refractory period \( D \) in any delay-advance phase response function (as exemplified in (2.2) and shown in Fig. 2.2). If the containing arc of the oscillators is less than some \( \bar{\Lambda} \in (0, \frac{1}{2}] \) and the refractory period \( D \) is not greater than \( 1 - \bar{\Lambda} \), then a strongly connected network of such oscillators using phase jumps will synchronize with each oscillator having independent and time-varying \( \alpha \in (0, 1] \) that does not converge to zero at firing instances that cause a non-zero phase adjustment.

**Proof.** Let us consider a PCO network where the initial phases are within some containing arc \( \Lambda < \bar{\Lambda} \). Without loss of generality, let us assume that oscillator \( i \) has the largest initial phase, \( \phi_{max} \) at time \( t = 0 \), oscillator \( j \) has the smallest initial phase such that \( \phi_j = \phi_{max} - \Lambda \), and all other oscillator phases reside between oscillators \( i \) and \( j \).

Since oscillator \( i \) has the largest phase, its phase evolves to 1 without perturbation and it
reaches the threshold at \( t = \frac{1 - \phi_{max}}{\omega_0} \). At this firing instance, all of the other oscillators have phases between \( 1 - \Lambda \) (which is larger than \( \frac{1}{2} \)) and 1. In the following time interval of length \( \frac{\Lambda}{\omega_0} \), every oscillator will fire once. Since the network is strongly connected, oscillator \( j \) receives at least one pulse during its phase evolution from \( 1 - \Lambda \) to 1, and its phase is increased. (The value of the phase response curve is positive in the interval \( (\frac{1}{2}, 1) \).) We denote the phase increase at that firing instance as \( \psi_j \), which is strictly positive and dependent on the time-varying coupling strength, \( \alpha \), and the phase response curve, and hence is time-dependent. Note that, since the sequence of values of the coupling strength at firing instances does not converge to zero, the phase increase \( \psi_j \) will also
not converge to zero unless \( \phi_j \), as in (2.1), converges to zero as the network approaches synchrony.

Given that the initial phase difference is \( \Lambda \), and that the phase of oscillator \( j \) is increased by \( \psi_j \),
the containing arc of the network decreases by at least \( \psi_j \), as oscillator \( i \) may have decreased its
phase due to the pulse received while in the interval \( [D, \Lambda) \), if \( D < \Lambda \) holds. (The value of the phase response curve is negative in the interval \( (0, \frac{1}{2}] \).) The network then continues on to the next cycle, and the above analysis repeats.
Therefore, since the containing arc of the network decreases with every cycle, and cannot be negative, then the containing arc converges to zero, and the network synchronizes. 

\[ \text{Remark 3. The coupling strengths of the oscillators can vary independently from each other, and synchronization will still occur, as long as the coupling strength is within } (0, 1) \text{ and such that the sequence of coupling strength values at firing instances that cause a non-zero adjustment in the oscillator’s phase does not converge to zero for each oscillator.} \]

Theorem 1 proves that a PCO network can synchronize for any potentially time-varying coupling strength, \( \alpha \). As shown in Sec. 3.1, phase continuity can be modeled as a reduction in the coupling strength of an oscillator as in (3.1). Since the bound of this effective coupling strength is \((0, \alpha]\), the effective coupling strength will also be inside the bound \((0, 1]\). Thus, we have the following theorem.

\[ \text{Theorem 2. Consider pulse-coupled oscillators with non-zero refractory period } D \text{ in any delay-advance phase response function (as exemplified in (2.2) and shown in Fig. 2.2). If the containing arc of the oscillators is less than some } \bar{\Lambda} \in (0, \frac{1}{2}] \text{ and the refractory period } D \text{ is not greater than } 1 - \bar{\Lambda}, \text{ then a strongly connected network of such PCOs with phase continuity as described in Sec. 3.1 will synchronize for } \alpha \in (0, 1]. \]

\[ \text{Proof. This proof follows from Theorem 1. Phase continuity results in individual oscillator coupling strengths, } \alpha_e, \text{ being independent, time-varying, and bounded within the interval } (0, \alpha], \text{ as shown in the analysis of Sec. 3.1. Thus, we must show that the sequence of coupling strength values does not converge to zero at firing instances that cause a non-zero phase adjustment for each oscillator. It is sufficient, then, to show that a subsequence of those values does not converge to zero.} \]

Consider the last pulse that an oscillator receives in a cycle that results in a phase adjustment. (As in Theorem 1, oscillator \( i \) fires first in the cycle, and oscillator \( j \) fires last in the cycle.) We will consider the two cases when the last received pulse occurs before the oscillator fires its own pulse, and when the last received pulse occurs after the oscillator fires its own pulse.

1. In the case when the last pulse that an oscillator receives occurs before the oscillator has fired its own pulse (e.g., oscillator \( j \)), since the oscillator will not respond to its own pulse (c.f. Remark 2), the oscillator will continue to respond to the last pulse it received for the current cycle. Thus, the effective coupling strength for that received pulse is given by (3.1), where \( t_0 \)
is the time difference between the last pulse received in the current cycle and the first pulse received in the next cycle.

2. In the case when the last pulse that an oscillator receives occurs after the oscillator has fired its own pulse (e.g., oscillator \( i \)), we only need to consider the case when the oscillator’s phase is in the interval \([D, \frac{1}{2}]\) if \( D \leq \frac{1}{2} \) holds. Any pulses received in the refractory period \([0, D]\) are ignored by the oscillator, resulting in no phase adjustment, and thus do not affect the behavior of the network. If the oscillator’s phase is in the interval \([D, \frac{1}{2}]\), given \( D \leq \frac{1}{2} \), the oscillator will respond to the pulse by adjusting its phase. Thus, the effective coupling strength for that received pulse is given by (3.1), where \( t_0 \) is the time difference between the last pulse received in the current cycle and the first pulse received in the next cycle.

For both cases, since the phases of all oscillators are contained within \( \bar{\Lambda} \in (0, \frac{1}{2}] \), then \( t_0 > \frac{1}{2\omega_0} \) holds for all oscillators (resulting in the effective coupling strength to be non-zero) and the containing arc will decrease as described in Theorem 1 during that cycle. As the network synchronizes, for any oscillator \( j \), the effective coupling strength for the last received pulse in the cycle will converge to the value \( \min \left\{ \frac{\omega}{\tau_j} \alpha, \alpha \right\} \neq 0 \), where \( \tau_j \) is finite, and where \( t_0 \) approaches the length of the cycle (i.e., \( \frac{1}{2\omega_0} \)).

Thus, for each oscillator, the subsequence of coupling strength values corresponding to the last received pulse that results in a non-zero phase adjustment during each cycle does not converge to zero, which implies that the entire sequence of coupling strength values does not converge to zero at each firing instance that results in a non-zero phase adjustment for each oscillator.

Therefore, the conditions for Theorem 1 are met, and the network converges to the state of synchronization.

Additional phase continuity methods besides the ones described in Chapter 2 may also be used to achieve continuous oscillator phase evolution. The concept of a time-varying effective coupling strength that is reduced from the network coupling strength is independent of the specific method chosen to ensure phase continuity.

3.2.2 Synchronization Time

As found in [55], the time to synchronization of the oscillator network using phase jumps is proportional to the containing arc of the oscillators and inversely proportional to the coupling
strength, average phase response, and the indegree of the network topology. We can show that the
time that a PCO network needs to synchronize under phase continuity is proportional to a similar
expression.

**Theorem 3.** Consider pulse-coupled oscillators having continuous phase using any delay-advance
phase response function with non-zero refractory period $D$ (as exemplified in (2.2) and shown in Fig.
2.2) in a strongly connected network such that the containing arc of the oscillators $\Lambda$ is less than
some $\bar{\Lambda} \in (0, \frac{1}{2}]$ and the refractory period $D$ not greater than $1 - \bar{\Lambda}$. Then the time to synchronization
is proportional to

$$\frac{\Lambda}{\alpha_e \Psi \delta^{-}(G)}$$

(3.2)

where $\alpha_e = \frac{1}{\delta^{-}(G)} \sum \alpha_e_i$ is the average effective coupling strength at firing instances of a single
oscillator during a cycle, $\delta^{-}(G)$ is the indegree of the network represented by graph $G$, and $\Psi$
denotes the average phase response of oscillators in the advance stage of the phase response function
(i.e., $[\frac{1}{2}, 1]$), given by

$$\Psi = 2 \int_{\frac{1}{2}}^{1} Q(\phi) p(\phi) d\phi$$

(3.3)

where $Q(\phi)$ denotes the value of the phase response function when the phase is $\phi$, and $p(\phi)$ denotes
the probability that an oscillator receives a pulse at phase $\phi$.

**Proof.** From Theorem 2, we know that the containing arc $\Lambda$ decreases by at least amount $\psi_j$ every
cycle, so the time to synchronization is proportional to the size of the initial containing arc $\Lambda$, and
is inversely proportional to the amount that the containing arc decreases, $\psi_j$. $\psi_j$ represents the
increase in the phase of the oscillator that fired last in a cycle, and is proportional to the sum of the
product of the effective coupling strength, $\alpha_e$, and the average phase increase in the advance stage
of the phase response function caused by a single pulse, given by $\Psi$ in (3.3), for all pulses received
in a cycle. The number of pulses received by an oscillator is defined to be the indegree of the vertex
 corresponding to that oscillator, which is at least $\delta^{-}(G)$. Both $\Psi$ and $\delta^{-}(G)$ can be factored from
this sum, leaving the expression for the average effective coupling strength, $\alpha_e$. Thus, we know that
the time to synchronization is determined by (3.2).

The result of Theorem 3 is similar to that found in [55]. However, due to the condition of
phase continuity, the time to synchronization is now inversely proportional to the average effective
coupling strength, rather than the coupling strength directly. When considering phase continuity, the
effective coupling strength will be less than or equal to the actual coupling strength of the network, implying that phase continuity will cause the network to converge more slowly. Furthermore, as the network synchronizes, the effective coupling strengths at firing instances for oscillator \( j \), as in Theorem 2, will approach zero for all but the last pulse that is received in every cycle. Thus, the sum of the effective coupling strengths will approach the value of the effective coupling strength from the last pulse received, and the dependence on the indegree of the network will disappear, increasing the total time to synchronization.

3.3 Simulations and Experimental Results

To test the results concerning phase continuity in pulse-coupled oscillator networks, we perform multiple simulations and physical experiments to observe the behavior of standard PCO networks and synchronization algorithms when phase continuity methods, as in Chapter 2, are applied. We first simulate the proposed phase continuity methods in MATLAB, and then test them on a physical PCO network of Raspberry Pi 3 Model B microcomputers.

3.3.1 PCO Simulations in MATLAB

We first perform simulations of PCO synchronization algorithms in MATLAB, using the two phase continuity methods as discussed in Chapter 2. Each oscillator evolves over the interval \([0, 1)\), with fundamental frequency \( \omega_0 = 1 \), and period of one second.

3.3.1.1 PRC Synchronization

We first use the PRC shown in Fig. 2.1 and expressed in (2.2). We first consider a PCO network under an all-to-all connection topology with a small refractory period \( D = 0.001 \). Fig. 3.1 shows that the network does indeed synchronize under both proposed phase continuity methods from Chapter 2. As expected, the network converges more slowly when under the phase continuity methods, due to the reduced effective coupling strengths of the oscillators at most firing instances. The time to synchronization is illustrated in Fig. 3.2 where the length of the containing arc is plotted as a function of time.

Next, we illustrate in Fig. 3.3 that the network with a large refractory period \( D = 0.5 \) will synchronize using the phase continuity methods in Chapter 2. As can be seen in Fig. 3.4,
Figure 3.1: Phase evolution of a PCO network using the PRC given in (2.2) for $N = 6$ oscillators in an all-to-all topology, with $\alpha = 0.5$, refractory period $D = 0.001$, and random initial conditions in a containing arc $\Lambda < \frac{1}{2}$. (a) Continuous phase evolution under the constant frequency method, with $\omega_a = 0.3\omega_0$. (b) Continuous phase evolution under the constant time method, with $\tau = 0.3$ seconds. (c) Phase jumps, for comparison to the phase continuity methods.

convergence is slower than when we used a small refractory period in Fig. 3.2. Again, the reduced coupling strength of the network due to phase continuity leads to an increased synchronization time compared to the phase jump case.

The PRC in (2.2) can also achieve synchronization in networks with a more generally connected topology. We illustrate this case in Fig. 3.5 and Fig. 3.7, where we use the bidirectional ring topology and the bidirectional line topology respectively, each network having a small refractory period $D = 0.001$. Again, we see in Fig. 3.6 and Fig. 3.8 that the use of the phase continuity methods in Chapter 2 still allows both networks to synchronize, although the convergence rate is decreased due to the reduced effective coupling strength of the oscillators at most firing instances.
Figure 3.2: Containing arcs, Λ, as a function of time for the networks in Fig. 3.1. The convergence speed of the containing arcs under the constant frequency method, with $\omega_a = 0.3\omega_0$, and the constant time method, with $\tau = 0.3$ seconds, is reduced compared with the phase jump case.

Theorem 2 requires the initial phases of the oscillators to be within a containing arc less than one half of the cycle to guarantee synchronization. As shown in [36], oscillators using phase jumps under certain topologies are guaranteed to synchronize from any initial starting phase with a strong enough coupling strength (e.g., $\alpha > 0.5$ for an all-to-all topology). Since the effective coupling strength of the oscillators is potentially reduced due to the constraint of phase continuity, it is impossible to guarantee an always strong effective coupling strength, so no strong statements paralleling those found in [36] can be made regarding synchronization from every initial condition (cf. Fig. 3.9 exemplifying a non-synchronizing case under $\alpha > 0.5$ and the all-to-all topology).

However, our simulation results show that synchronization is still very likely to occur when the phases are randomly distributed across the entire cycle. More specifically, after 250 simulations for $N = 6$ oscillators in an all-to-all topology, with $\alpha = 0.51$, refractory period $D = 0.001$, and initial phases randomly distributed from $[0,1)$ such that the containing arc $\Lambda > \frac{1}{2}$ holds, we found that more than 97% of the cases still resulted in synchronization under both proposed phase continuity methods ($\omega_a = 0.3\omega_0; \tau = 0.3$ seconds). Note that in [36], where phase continuity is not considered, all initial conditions will synchronize for the given coupling strength.
Figure 3.3: Phase evolution of a PCO network using the PRC given in (2.2) for $N = 6$ oscillators in an all-to-all topology, with $\alpha = 0.5$, refractory period $D = 0.5$, and random initial conditions in a containing arc $\Lambda < \frac{1}{2}$. (a) Continuous phase evolution under the constant frequency method, with $\omega_a = 0.3\omega_0$. (b) Continuous phase evolution under the constant time method, with $\tau = 0.3$ seconds. (c) Phase jumps, for comparison to the phase continuity methods.

3.3.1.2 Peskin Synchronization Algorithm

We will next consider the original PCO model that was first introduced by Peskin in [42]. He described the oscillators as “integrate-and-fire” oscillators, increasing in phase and firing and resetting their phase when they reached a threshold. The phase of an oscillator is mapped onto a state variable, $x_i(t)$, using the relation $x_i(t) = f(\phi_i(t))$, where $f(\phi)$ is a function that is “smooth, monotonic increasing, and concave down” [26]. When a pulse is received, the oscillator maps its current phase to the state variable, increments the state variable by an amount $\epsilon$, and then maps the state back to the phase using the inverse function $g(x) = f^{-1}(x)$. That is, the new phase of the
oscillator can be written as

$$\phi_i(t) = g(f(\phi_i(t)) + \epsilon) \quad (3.4)$$

If the state variable is incremented past the threshold value (i.e., $f(\phi_i(t)) + \epsilon > 1$), then the oscillator immediately fires and resets its phase to zero, and becomes completely synchronized with the oscillator that had fired previously.

Any function $f(\phi)$ that meets the requirements as above can be used to map the phase into the state variable. Peskin used the following function and its associated inverse:

$$f(\phi) = (1 - e^{-\gamma})(1 - e^{-\gamma\phi}) \quad (3.5)$$

$$g(x) = \frac{1}{\gamma} \ln\left(\frac{1 - e^{-\gamma}}{1 - e^{-\gamma} - x}\right) \quad (3.6)$$

Mirrollo and Strogatz further improved Peskin’s model in [26], and used an alternate function for mapping the phase to the state variable:

$$f(\phi) = \frac{1}{b} \ln(1 + (e^b - 1)\phi) \quad (3.7)$$
Figure 3.5: Phase evolution of a PCO network using the PRC given in (2.2) for $N = 6$ oscillators in a ring topology, with $\alpha = 0.5$, refractory period $D = 0.001$, and random initial conditions in a containing arc $\Lambda < \frac{1}{2}$. (a) Continuous phase evolution under the constant frequency method, with $\omega_a = 0.3\omega_0$. (b) Continuous phase evolution under the constant time method, with $\tau = 0.3$ seconds. (c) Phase jumps, for comparison to the phase continuity methods.

\[ g(x) = \frac{e^{bx} - 1}{e^b - 1} \]  

(3.8)

Using these functions, an equivalent phase response curve (PRC) can be found by determining $\phi_i$ from (2.1). Fig. 3.10 illustrates these equivalent PRCs.

It is important to note that the Peskin PCO model does not incorporate any kind of coupling strength parameter, $\alpha$, as in the PRC synchronization given in [55]. This is easily verified by noting that the state variable increment $\epsilon$ does not simply scale the equivalent PRC function, but modifies the overall shape of the function. The Peskin model requires that the phases jump the entire
amount necessary according to the state mapping function parameters. Equivalently, the Peskin model assumes that the coupling strength $\alpha$ for the network is always 1.

The lack of a coupling strength parameter makes it difficult to extend the analysis of the previous sections to the Peskin model. However, even though there is no coupling strength inherent to the Peskin model, simulations show good synchronization results when the phase continuity methods are applied. For example, using the state variable created from the functions given in (3.5) and (3.6), and the equivalent PRC function, we find the amount that the oscillator would jump and apply phase continuity methods from Chapter 2. Fig. 3.11 shows that the phase continuity methods still allow the PCO network to synchronize.

Similarly, using the alternate state variable function introduced by Mirollo and Strogatz given in (3.7) and (3.8) also allows the network to synchronize under the phase continuity methods, as shown in Fig. 3.12.

### 3.3.1.3 Reachback Firefly Algorithm

Another synchronization algorithm that has been proposed is the Reachback Firefly Algorithm (RFA) by Werner-Allen et. al. in [57]. This algorithm is based on the Peskin PCO model,
Figure 3.7: Phase evolution of a PCO network using the PRC given in (2.2) for $N = 6$ oscillators in a line topology, with $\alpha = 0.5$, refractory period $D = 0.001$, and random initial conditions in a containing arc $\Lambda < \frac{1}{2}$. (a) Continuous phase evolution under the constant frequency method, with $\omega_a = 0.3\omega_0$. (b) Continuous phase evolution under the constant time method, with $\tau = 0.3$ seconds. (c) Phase jumps, for comparison to the phase continuity methods. 

where the phase is mapped to a state variable. RFA uses a simple mapping function to decrease computational complexity.

\[ f(\phi) = \ln(\phi) \]  

\[ g(x) = e^x \]  

Fig. 3.10 also illustrates the equivalent PRC function for the functions used in the RFA model.

The key difference between the RFA model and the Peskin model is that the oscillators wait
Figure 3.8: Containing arcs, $\Lambda$, as a function of time for the networks in Fig. 3.7. The convergence speed of the containing arcs under the constant frequency method, with $\omega_a = 0.3\omega_0$, and the constant time method, with $\tau = 0.3$ seconds, is reduced compared with the phase jump case.

to jump until the moment they fire. As the oscillator receives pulses, it records how much it would jump, according to the state variable mapping, at each time instance. When the oscillator reaches the threshold and fires, it then adds all of the recorded jump amounts from the previous cycle, and jumps by the total amount. This process is then repeated for each cycle until synchronization is achieved.

As with the Peskin model, there is no inherent coupling strength parameter $\alpha$ in the RFA model. The coupling strength $\alpha$ is assumed to be always 1. This lack of coupling strength incorporated into the RFA model makes it similarly difficult to extend the analysis from the previous sections. However, like with the Peskin model, simulations show good synchronization results when the phase continuity methods are applied. Fig. 3.13 illustrates that the phase continuity methods in Chapter 2 allow the PCO network to synchronize. It is also important to note that the phase continuity method parameters can be within a broader interval than for the simpler Peskin model. Since each oscillator does not jump when it receives a pulse, and only when it fires, the effective coupling strength can be maximized more easily than in the Peskin model.
Figure 3.9: Containing arcs, Λ, as a function of time in a network that does not converge for $N = 6$ oscillators in an all-to-all topology. The coupling strength was $\alpha = 0.51$ and the refractory period was $D = 0.001$. The initial conditions were randomly selected, resulting in a containing arc of length $\Lambda > \frac{1}{2}$. Under the constant frequency method ($\omega_a = 0.3 \omega_0$), the oscillators were unable to synchronize, but under the constant time method ($\tau = 0.3$ seconds), the oscillators achieved synchronization.

### 3.3.2 PCO Experiments on Raspberry Pi

We next perform physical experiments of PCO synchronization algorithms on a network of Raspberry Pi 3 Model B microcomputers, using the two phase continuity methods discussed in Chapter 2. These physical experiments verify the simulations done in MATLAB, while accounting for non-ideal conditions found in physical PCO networks, such as propagation delay of transmitted pulses and variations in fundamental frequency [21]. Each Raspberry Pi models a single oscillator and sends pulses to other Raspberry Pis in the network using an Xbee RF communication module, as shown in Fig. 3.14. Each Raspberry Pi runs an identical set of code written in Python, and the phase of each oscillator is based on the internal clock of the Raspberry Pi and evolves over the interval $[0, 1)$. 

32
3.3.2.1 PRC Synchronization

We first use the PRC shown in Fig. 2.1 and expressed in (2.2) under an all-to-all connection topology with a small refractory period $D = 0.001$. The period of one cycle is set to be ten seconds, which leads to a fundamental frequency $\omega_0 = 0.1$ for each oscillator. We give the oscillators in the network the same initial phase and phase continuity parameters as used in the MATLAB simulations to compare the behavior between the two networks. Fig. 3.15 shows that the physical network does indeed synchronize. As expected, the network synchronizes more slowly when under the phase continuity methods from Chapter 2, due to the reduced effective coupling strengths of the oscillators at most firing instances. The rate of convergence is illustrated in Fig. 3.16 where the length of the containing arc is plotted as a function of time.

Next, we illustrate in Fig. 3.17 that the physical network with a large refractory period $D = 0.5$ will still synchronize using the phase continuity methods in Chapter 2. As can be seen in Fig. 3.18, convergence is slower than when we excluded the refractory period in Fig. 3.16. Again, the reduced coupling strength of the network due to phase continuity leads to an increased synchronization time compared to the phase jump case.

The PRC in (2.2) can also achieve synchronization in networks with a more generally con-
Figure 3.11: Containing arcs, $\Lambda$, as a function of time for the networks under the Peskin synchronization model with parameters $\epsilon = 0.002, \gamma = 3$. The convergence property of the containing arcs under the constant frequency method, with $\omega_a = 0.3\omega_0$, and the constant time method, with $\tau = 0.1$ seconds, is maintained compared to the phase jump case.

3.3.2.2 Peskin Synchronization Algorithm

We next reconsider the Peskin synchronization model, and perform experiments to see how the physical PCO network responds under non-ideal conditions and phase continuity. We use the same initial phase values and parameters for the Peskin state variable and phase update functions as used in the MATLAB simulations to compare the response of both networks. That is, the oscillators evolve over the interval $[0,1)$, with fundamental frequency $\omega_0 = 1$, and period of one second.
Figure 3.12: Containing arcs, Λ, as a function of time for the networks under the Mirollo-Strogatz synchronization model with parameters $\epsilon = 0.002$, $b = 5$. The convergence property of the containing arcs under the constant frequency method, with $\omega_a = 0.3\omega_0$, and the constant time method, with $\tau = 0.1$ seconds, is maintained compared to the phase jump case.

Again, we see that we get good synchronization results under phase continuity, even though the synchronization algorithm does not incorporate a coupling strength parameter, as discussed previously. Fig. 3.23 shows that the phase continuity methods still allow the physical PCO network to synchronize. Similarly, using the alternate state variable function introduced by Mirollo and Strogatz also allows the network to synchronize under the phase continuity methods, as shown in Fig. 3.24.

An important difference between the synchronization of the simulations and the physical PCO network under the Peskin synchronization algorithm should be addressed. Specifically, the jump case in the MATLAB simulation synchronizes much more slowly than in the physical network, although oscillators under the phase continuity methods synchronize in comparable amounts of time. This behavior is due to the propagation delay of the transmitted pulse between oscillators. Ideally, once two oscillators synchronize, they maintain the same phase over time and form a single oscillator group. Any oscillators in an oscillator group will fire together, forming a single transmitted pulse, resulting in fewer pulses sent per cycle until all oscillators are synchronized, eventually resulting in one pulse per cycle. However, in a physical PCO network, these oscillator groups cannot form perfectly, so each oscillator receives the same number of pulses in each cycle, allowing them to
Figure 3.13: Containing arcs, $\Lambda$, as a function of time for the networks under the Reachback Firefly Algorithm (RFA) with parameter $\epsilon = 0.002$. The convergence property of the containing arcs under the constant frequency method, with $\omega_a = 0.007\omega_0$, and the constant time method, with $\tau = 1.1$ seconds, is maintained compared to the phase jump case. The RFA algorithm further allows for phase continuity parameters to be in a broader interval than the Peskin algorithm.

synchronize more quickly. For oscillators operating under phase continuity, the increase in the number of pulses received is countered by a decrease in the effective coupling strength.
Figure 3.15: Phase evolution of a physical PCO network using the PRC given in (2.2) for \( N = 6 \) oscillators in an all-to-all topology, with \( \alpha = 0.5 \), refractory period \( D = 0.001 \), and random initial conditions in a containing arc \( \Lambda < \frac{1}{2} \). (a) Continuous phase evolution under the constant frequency method, with \( \omega_0 = 0.3\omega_0 \). (b) Continuous phase evolution under the constant time method, with \( \tau = 3 \) seconds. (c) Phase jumps, for comparison to the phase continuity methods.
Figure 3.16: Containing arcs, $\Lambda$, as a function of time for the networks in Fig. 3.15. The convergence speed of the containing arcs under the constant frequency method, with $\omega_n = 0.3\omega_0$, and the constant time method, with $\tau = 3$ seconds, is reduced compared with the phase jump case.
Figure 3.17: Phase evolution of a physical PCO network using the PRC given in (2.2) for $N = 6$ oscillators in an all-to-all topology, with $\alpha = 0.5$, refractory period $D = 0.5$, and random initial conditions in a containing arc $\Lambda < \frac{1}{2}$. (a) Continuous phase evolution under the constant frequency method, with $\omega_a = 0.3\omega_0$. (b) Continuous phase evolution under the constant time method, with $\tau = 3$ seconds. (c) Phase jumps, for comparison to the phase continuity methods.
Figure 3.18: Containing arcs, $\Lambda$, as a function of time for the networks in Fig. 3.17. The convergence speed of the containing arcs under the constant frequency method, with $\omega = 0.3\omega_0$, and the constant time method, with $\tau = 3$ seconds, is reduced compared with the phase jump case.
Figure 3.19: Phase evolution of a physical PCO network using the PRC given in (2.2) for $N = 6$ oscillators in a ring topology, with $\alpha = 0.5$, refractory period $D = 0.001$, and random initial conditions in a containing arc $\Lambda < \frac{1}{2}$. (a) Continuous phase evolution under the constant frequency method, with $\omega_a = 0.3 \omega_0$. (b) Continuous phase evolution under the constant time method, with $\tau = 0.3$ seconds. (c) Phase jumps, for comparison to the phase continuity methods.
Figure 3.20: Containing arcs, Λ, as a function of time for the networks in Fig. 3.19. The convergence speed of the containing arcs under the constant frequency method, with $\omega_a = 0.3\omega_0$, and the constant time method, with $\tau = 0.3$ seconds, is reduced compared with the phase jump case.
Figure 3.21: Phase evolution of a physical PCO network using the PRC given in (2.2) for $N = 6$ oscillators in a line topology, with $\alpha = 0.5$, refractory period $D = 0.001$, and random initial conditions in a containing arc $\Lambda < \frac{1}{2}$. (a) Continuous phase evolution under the constant frequency method, with $\omega_a = 0.3\omega_0$. (b) Continuous phase evolution under the constant time method, with $\tau = 0.3$ seconds. (c) Phase jumps, for comparison to the phase continuity methods.
Figure 3.22: Containing arcs, Λ, as a function of time for the networks in Fig. 3.21. The convergence speed of the containing arcs under the constant frequency method, with $\omega_a = 0.3\omega_0$, and the constant time method, with $\tau = 0.3$ seconds, is reduced compared with the phase jump case.

Figure 3.23: Containing arcs, Λ, as a function of time for the networks under the Peskin synchronization model with parameters $\epsilon = 0.002, \gamma = 3$. The synchronization of the network under the constant frequency method, with $\omega_a = 0.3\omega_0$, and the constant time method, with $\tau = 0.1$ seconds, is achieved more slowly when compared to the phase jump case.
Figure 3.24: Containing arcs, Λ, as a function of time for the networks under the Mirollo-Strogatz synchronization model with parameters $\epsilon = 0.002$, $b = 5$. The synchronization of the network under the constant frequency method, with $\omega_a = 0.3\omega_0$, and the constant time method, with $\tau = 0.1$ seconds, is achieved more slowly when compared to the phase jump case.
Chapter 4

Decentralized Heading Control under Rate Constraints

In this chapter, we consider the task of applying the pulse-coupled oscillator model to control the heading of mobile robot networks. In Section 4.1, we will describe the model for mobile robots under heading rate constraints and its relationship to oscillators under phase rate constraints. We will then rigorously show how a mobile robot network under heading rate constraints can achieve heading synchronization and desynchronization in Sections 4.2 and 4.3, respectively. Next, we will present physical experiments on a robotic platform in Section 4.4 that demonstrate the application of pulse-coupled oscillator strategies under the heading rate constraint to collective motion coordination. Throughout this chapter, we will use the phase threshold value $c_\phi = 2\pi$.

4.1 Pulse-Based Heading Coordination

Let $\theta_i(t) \in S^1$ represent the heading of robot $i \in V$ in a network of $N$ identical robots. To control the heading of the mobile robot network, we propose the following PCO-inspired framework:\footnote{This analysis has been published in the IEEE Transactions on Control of Network Systems, Vol. 7, No. 3. [4].}

In addition to the variable $\theta_i(t)$ to represent the heading of robot $i$, an internal phase variable $\phi_i \in [0, 2\pi)$ is given to each robot such that its initial value is the initial heading of the robot, $\theta_i(0)$. This auxiliary variable represents the phase of an oscillator, and evolves at the rate, or frequency, $\omega_i$. Under no interactions between neighboring robots, the phase variable evolves at the fundamental
frequency $\omega_i = \omega_0$. When the value of the phase variable $\phi_i$ reaches the threshold of $2\pi$, it resets to zero, and the robot sends out, or fires, a pulse to all robots connected to it in the network.

The control for the heading of the robot, then, follows the phase variable. The rate at which the robot rotates, $\dot{\theta}_i$, is given by

$$\dot{\theta}_i = \dot{\phi}_i - \omega_0$$

(4.1)

where $\dot{\phi}_i = \omega_i$ is the rate at which the phase variable evolves.

However, there is a physical constraint on how fast the robot can rotate. That is, $|\dot{\theta}_i| \leq \omega_{\text{max}}$ must hold for some maximum rotational rate $\omega_{\text{max}}$. This natural constraint requires that the heading, $\theta_i(t)$, evolves continuously on the one-dimensional torus. Thus, a constraint is implied on the rate of the phase variable $\phi_i$, such that

$$|\dot{\theta}_i| = |\dot{\phi}_i - \omega_0| \leq \omega_{\text{max}}$$

(4.2)

holds. This implied constraint on the phase variable, $\phi_i$, also requires it to evolve continuously on the one-dimensional torus.

To our knowledge, all existing literature on PCO networks has the phase value change instantaneously, i.e. $|\dot{\phi}| = \infty$, at pulse firing instances. Since the heading rate constraint implies a rate constraint on the auxiliary phase, we require that the phase not change instantaneously. To ensure that the phase evolves continuously, we will utilize a phase continuity method as given in Chapter 2. Specifically, we will use the constant frequency method, where we use $\omega_a = \omega_{\text{max}}$ to ensure the heading rate constraint in (4.2).

Therefore, when a robot receives a pulse, its phase variable will adjust its rate according to the constant frequency phase continuity method described in Chapter 2. That is, using the chosen phase response function $F(\phi_i)$, the phase variable will begin to evolve at rate $\omega_i$ for time $\tau_i$, and then return to evolving at the fundamental frequency $\omega_0$. Thus, based on the heading control given in (4.1), during the normal evolution of the phase variable, the robot does not rotate, and when a pulse is received, the robot will rotate at rate $\dot{\theta}_i = \pm \omega_{\text{max}}$ for time $\tau_i$ and then stop rotating. If a new pulse is received before time $\tau_i$ is complete, then the rate of rotation for the robot will update for the new amount of time $\tau_i$. Fig. 4.1 illustrates the relationship between the robot’s phase variable and heading in a network with six robots in an all-to-all topology.

With the heading control given in (4.1), as the values for the phase variables across all
of the robots in the network achieve their desired state by following the chosen PRC, the heading of the mobile robots will also achieve the same desired state. It leaves us to show, then, that the phase variables in the network are able to achieve the desired states of synchronization and desynchronization under the heading rate constraint.

4.2 **Synchronization under Heading Rate Constraint**

In this section, we consider the dynamics of synchronizing mobile robots under rate-constrained heading evolution. All that is required is to take the amount of adjustment for phase variable $\phi_i$, i.e. $\psi_i$, and determine the necessary amount of time, $\tau_i$, to change the frequency $\omega_i$. We then let the phase variable evolve at the new frequency for the required amount of time before it returns to its fundamental frequency, $\omega_0$.

4.2.1 **Phase Response Function**

We will consider the pulse-coupled oscillator PRC synchronization strategy given in [55], the same as used in Chapter 3. We will use the delay-advance phase response curve, $Q$, given by (2.2), where we use $c_\phi = 2\pi$, with a non-zero refractory period $D$. When an oscillator receives a
pulse, it updates its phase variable according to $Q$ as shown in Fig. 2.2.

### 4.2.2 Synchronization with Heading Rate Constraint

As shown in Chapter 3, the PRC in (2.2) is capable of achieving synchronization in a connected PCO network under the implied phase rate constraint as described in Sec. 4.1, where it is possible to model the behavior of the network using a time-varying coupling strength for each phase variable in the network. Specifically, if a new pulse arrives at time $t_0 < \tau_i$, then the phase variable has only achieved a fraction of the amount of the desired phase change before it responds to the new pulse. This fractional amount of the desired phase change can be seen as a reduction of the coupling strength, $\alpha$, of robot $i$ by the ratio $\frac{t_0}{\tau_i}$, and thus the effective coupling is bounded in the interval $(0, \alpha]$.

We can apply this result to the control of mobile robots.

**Theorem 4.** Let $N$ mobile robots, each with heading $\theta_i$, be in a (strongly) connected network topology, such that the containing arc of the headings is less than some $\bar{\Lambda} \in (0, \pi]$. Then using the phase response curve given in (2.2) with $\alpha \in (0, 1]$ and refractory period $D$ not greater than $2\pi - \bar{\Lambda}$, the heading control given in (4.1) will synchronize the headings of the robots with the heading rate constraint given in (4.2).

*Proof.* The proof for this theorem follows from the results found in Chapter 3. By using the heading control given in (4.1) such that the initial value for the phase variable $\phi_i$ is set to be $\theta_i(0)$, we have that $\theta_i = (\phi_i - \omega_0 t) \mod 2\pi$ for $i \in \mathcal{V}$. Thus, to show that the headings of the robots $\theta_1, \ldots, \theta_N$ achieve the state of synchronization, we only need to show that the phase variables $\phi_1, \ldots, \phi_N$ achieve the state of synchronization.

Since the initial headings of the robots are in a containing arc $\bar{\Lambda} \leq \pi$, the initial phase variable values will also be in a containing arc $\bar{\Lambda} \leq \pi$. From the results in Chapter 3, the phase variables following the phase response curve given in (2.2) under the implied phase rate constraint given in (4.2) will synchronize for $\alpha \in (0, 1]$ and refractory period $D$ not greater than $2\pi - \bar{\Lambda}$. Therefore, the headings of the robots $\theta_1, \ldots, \theta_N$ will achieve synchronization.

This result allows for the heading of the mobile robots to synchronize while conforming to the heading rate constraint. Using previous results on pulse-coupled oscillator synchronization, the headings of the robots would have to adjust instantaneously to achieve synchronization, which is not
physically possible or desirable. The only effect of the heading rate constraint is a slight decrease in the convergence speed of the network toward synchronization, as shown by Theorem 3 in Chapter 3, due to the effective decrease in coupling strength.

4.3 Desynchronization under Heading Rate Constraint

We now rigorously analyze the dynamics of desynchronizing mobile robots with different initial headings under rate constrained heading evolution. As in the previous section on synchronization, all that is required is to take the amount of adjustment for phase variable $\phi_i$, i.e. $\psi_i$, and determine the necessary amount of time, $\tau_i$, to change the frequency $\omega_i$. We then let the phase variable evolve at the new frequency for the required amount of time before it returns to its fundamental frequency, $\omega_0$.

4.3.1 Phase Response Function

To analyze the network dynamics, we will use the phase response function, $F$, given by (2.6), where we use $c_\phi = 2\pi$, as shown in Fig. 4.2. Thus, from (2.1), a phase update rule can be
written as

\[ \phi_i^+ = \phi_i + F(\phi_i) = \begin{cases} 
(1 - l_1)\phi_i + l_1 \frac{2\pi}{N} & 0 < \phi_i < \frac{2\pi}{N} \\
\phi_i & \frac{2\pi}{N} \leq \phi_i \leq 2\pi \frac{N-1}{N} \\
(1 - l_2)\phi_i + l_2 2\pi \frac{N-1}{N} & 2\pi \frac{N-1}{N} < \phi_i < 2\pi 
\end{cases} \]  

(4.3)

From (4.3), \( \phi_i^+ \in (0, 2\pi) \) is a strictly monotonically increasing function of \( \phi_i \in (0, 2\pi) \), as illustrated with the solid blue line in Fig. 4.3. We will use this PRF as a basis for ensuring the phase rate constraint.

### 4.3.2 Effects of Rate Constraints on Phase Response Function

Let us analyze the behavior of the robot network under the heading rate constraint given in (4.2). Once robot \( i \) has received a pulse and calculated the necessary change in frequency, \( \omega_i \), to achieve the phase adjustment \( \psi_i \) in time \( \tau_i \), two possibilities can follow: 1) the robot receives no new pulses within time \( \tau_i \) that cause a phase adjustment, and 2) the robot receives a new pulse within time \( \tau_i \) that causes a phase adjustment. If we represent the duration of the time interval between the new and previous pulse firing instances as \( t_0 \), we can divide these two cases mathematically as 1) \( t_0 \geq \tau_i \), and 2) \( t_0 < \tau_i \).
1. In the first case, robot \( i \) finishes adjusting its phase variable by \( \psi_i \), and returns to evolving at the fundamental frequency \( \omega_0 \). The same effective change in the phase variable and robot heading has been achieved as if the robot had immediately changed its phase variable and heading by \( \psi_i \) and let the phase variable evolve at rate \( \omega_0 \) for a time of duration \( t_0 \). Thus, no effective change in the phase update rule occurs.

2. In the second case, the robot has not yet achieved its desired amount of change in the phase variable. Rather than having adjusted the whole amount \( \psi_i \), it has adjusted only a portion of that amount, \( \frac{t_0}{\tau_i} \psi_i \), in the time interval between received pulses. The robot then will use its current phase value at the time when the new pulse is received to redetermine new values for \( \psi_i \) and \( \tau_i \). This truncated phase evolution is equivalent to having the robot immediately change its phase variable and heading by \( \frac{t_0}{\tau_i} \psi_i \) and letting the phase variable evolve at rate \( \omega_0 \) for a time of duration \( t_0 \). This fractional amount of the desired change can be seen as a reduction of the coupling strength, \( l_1 \) or \( l_2 \), of the PRF for robot \( i \) by the ratio \( \frac{t_0}{\tau_i} \).

We now consider how (2.6) is affected by this behavior caused by the heading rate constraint given in (4.2).

Considering the constant frequency method, the maximum amount of time needed to make the desired phase adjustment is \( \tau_{\text{max}} = \max\{l_1, l_2\} \frac{2\pi}{\omega_{\text{max}} N} \). If, as in the first case above, there are no new pulses within time \( \tau_{\text{max}} \), then there is no effective change to the behavior of the network. Similarly, if the required time, \( \tau_i \), for robot \( i \) to complete the phase adjustment, \( \psi_i \), is smaller than \( t_0 \), then there is no effective change to the behavior of that robot. Otherwise, if the required phase adjustment, \( \psi_i \), is larger than \( \omega_{\text{max}} t_0 \) (i.e., if \( \tau_i > t_0 \)), then each robot with a phase variable \( \phi_i \in (0, \frac{2\pi}{N} - \frac{\omega_{\text{max}} t_0}{l_1}) \cup (\frac{2\pi}{N} + \frac{\omega_{\text{max}} t_0}{l_2}, 2\pi) \) when the previous pulse was fired only completes a phase adjustment of \( \omega_{\text{max}} t_0 \), a fractional amount of the required adjustment. Thus, for that previous pulse, we can describe the behavior of the network with a PRF given by

\[
F_\omega(\phi_i) = \begin{cases} 
L_1(\phi_i)(\frac{2\pi}{N} - \phi_i) & 0 < \phi_i < \frac{2\pi}{N} \\
0 & \frac{2\pi}{N} \leq \phi_i \leq 2\pi \frac{N-1}{N} \\
L_2(\phi_i)(\frac{2\pi}{N} - \phi_i) & 2\pi \frac{N-1}{N} < \phi_i < 2\pi 
\end{cases} \quad (4.4a)
\]

where the forward coupling function, \( L_1(\phi_i) \), and the backward coupling function, \( L_2(\phi_i) \), are given
by
\[
L_1(\phi_i) = \begin{cases} 
\frac{\omega_{\text{max}0}}{N} - \phi_i & 0 < \phi_i < \frac{2\pi}{N} - \frac{\omega_{\text{max}0}}{l_1} \\
l_1 & \frac{2\pi}{N} - \frac{\omega_{\text{max}0}}{l_1} \leq \phi_i < \frac{2\pi}{N} 
\end{cases} 
\] (4.4b)

and
\[
L_2(\phi_i) = \begin{cases} 
l_2 & 2\pi \frac{N-1}{N} < \phi_i \leq 2\pi \frac{N-1}{N} + \frac{\omega_{\text{max}0}}{l_2} \\
\frac{\omega_{\text{max}0}}{2\pi \frac{N-1}{N} - \phi_i} & \frac{2\pi}{N} - \frac{\omega_{\text{max}0}}{l_2} < \phi_i < 2\pi
\end{cases} 
\] (4.4c)
such that \( \frac{\omega_{\text{max}0}}{l_1} < \frac{2\pi}{N} \) and \( \frac{\omega_{\text{max}0}}{l_2} < \frac{2\pi}{N} \). This resulting PRF and the corresponding phase update rule are illustrated with red dotted lines in Fig. 4.2 and Fig. 4.3, respectively. Note that, in (4.4), \( L_1(\phi_i) \leq l_1 \) holds for \( \phi_i \in (0, \frac{2\pi}{N}) \), and \( L_2(\phi_i) \leq l_2 \) holds for \( \phi_i \in (2\pi \frac{N-1}{N}, 2\pi) \). Also, though the forward and backward coupling strengths, \( L_1(\phi_i) \) and \( L_2(\phi_i) \), are not constant over their respective domains, the resulting phase update rule, as illustrated in Fig. 4.3, is still strictly monotonically increasing.

**Remark 4.** The result of applying the heading rate constraint to (2.6) can be interpreted as a PRF with potentially reduced coupling strength functions that are dependent on the value of the phase variable, and that are time-varying based on the timing between received pulses.

**Remark 5.** The modification to the PRF due to the heading rate constraint described above does not actually adjust the coupling strengths as the network evolves. Only the apparent behavior of the network is being modeled as a PRF with reduced coupling strengths. The actual coupling strengths, \( l_1 \) and \( l_2 \), remain unchanged throughout the entire evolution of the network.

### 4.3.3 Analysis of Phase Response Functions with Time-Varying Coupling Strength Functions

We now desire to rigorously prove that a network operating under the time-varying phase response function of the form in (4.4), and thus under the heading rate constraint, can achieve desynchronization. To that end, we must prove some preliminary results.
Consider a PRF in the form

\[
F(\phi_i) = \begin{cases} 
L_1(\phi_i)(\frac{2\pi}{N} - \phi_i) & 0 < \phi_i < \frac{2\pi}{N} \\
0 & \frac{2\pi}{N} \leq \phi_i \leq 2\pi \frac{N-1}{N} \\
L_2(\phi_i)(2\pi \frac{N-1}{N} - \phi_i) & 2\pi \frac{N-1}{N} < \phi_i < 2\pi 
\end{cases}
\] (4.5)

and corresponding phase update rule

\[
\phi_i^+ = \phi_i + F(\phi_i) = \begin{cases} 
(1 - L_1(\phi_i))\phi_i + L_1(\phi_i)\frac{2\pi}{N} & 0 < \phi_i < \frac{2\pi}{N} \\
\phi_i & \frac{2\pi}{N} \leq \phi_i \leq 2\pi \frac{N-1}{N} \\
(1 - L_2(\phi_i))\phi_i + L_2(\phi_i)2\pi \frac{N-1}{N} & 2\pi \frac{N-1}{N} < \phi_i < 2\pi 
\end{cases}
\] (4.6)

such that the forward coupling function, \(L_1(\phi_i) \in [0, 1]\) over the domain \((0, \frac{2\pi}{N})\), and the backward coupling function, \(L_2(\phi_i) \in [0, 1]\) over the domain \((2\pi \frac{N-1}{N}, 2\pi)\), both of which may be time-varying, cause the phase update rule in (4.6) to be strictly monotonically increasing.

**Remark 6.** The PRF given in (2.6) can be written in the form of (4.5), where \(L_1(\phi_i) = l_1\) and \(L_2(\phi_i) = l_2\). Additionally, the PRF given in (4.4a) can be written in the form of (4.5), where \(L_1(\phi_i)\) and \(L_2(\phi_i)\) are given by (4.4b) and (4.4c) respectively.

We must first show that a robotic network under a PRF in the form of (4.5) can achieve phase desynchronization. Let us initially consider a robotic network that allows instantaneous change in the phase variable such that the coupling strength functions \(L_1(\phi_i)\) and \(L_2(\phi_i)\) are fixed with respect to time and only vary with respect to the value of the phase variable.

**Lemma 1.** For a network of \(N\) robots with no two robots having equal initial heading, and thus equal initial phase variables, the network will achieve phase desynchronization if the phase response function \(F(\phi_i)\) is given by (4.5) such that the forward coupling function, \(L_1(\phi_i) \in [0, 1]\), over the domain \((0, \frac{2\pi}{N})\) and the backward coupling function, \(L_2(\phi_i) \in [0, 1]\), over the domain \((2\pi \frac{N-1}{N}, 2\pi)\) cause the phase update rule in (4.6) to be strictly monotonically increasing.

**Proof.** The proof is given in Appendix A.
Remark 7. The only condition placed on the forward coupling function, \( L_1(\phi_i) \), and the backward coupling function, \( L_2(\phi_i) \), is that they cause the resulting phase update rule to be strictly monotonically increasing for \( \phi \in [0, 2\pi) \). Any pair of functions, either continuous or discontinuous, that satisfy this condition can be used to form a suitable phase response function.

Lemma 1 proves that a robotic network with a fixed PRF as in (4.5) can achieve phase desynchronization under instantaneous changes in the phase variable. We now use this result to show that a PRF that has forward and backward coupling functions that are time-varying, yet still cause the corresponding phase update rule in (4.6) to be strictly increasing over the interval \([0, 2\pi)\) can be used to achieve phase desynchronization.

Proposition 2. The evolution of the phase variable, and thus the heading, of a robot in a robotic network is dependent upon the value of the coupling strengths, \( L_1(\phi_i) \) and \( L_2(\phi_i) \), only at firing instances.

Proof. The proof for this proposition is straightforward. A robot only determines the amount that its phase variable needs to change when it receives a pulse. Thus, the values of the coupling strengths are only used at firing instances. Any values the coupling strengths take between firing instances are unused and thus independent of the behavior of the network.

Using the preliminary results of Lemma 1 and Proposition 2, we can prove the following theorem.

Theorem 5. For a network of \( N \) robots with no two robots having equal initial heading, and thus equal initial phase variables, the network will achieve phase desynchronization if the phase response function \( F(\phi_i) \) is given by (4.5) such that the forward coupling function, \( L_1(\phi_i) \in [0, 1) \), over the domain \((0, \frac{2\pi}{N})\) and the backward coupling function, \( L_2(\phi_i) \in [0, 1) \), over the domain \((2\pi - \frac{1}{N} \cdot \frac{2\pi}{N}, 2\pi)\) can vary over time, yet still cause the phase update rule in (4.6) to be strictly increasing over the entire interval \([0, 2\pi)\).

Proof. The proof for this theorem is straightforward. From Proposition 2, we only need to consider the phase response function at firing instances. At each firing instance, the phase response function fulfills the conditions for Lemma 1. Therefore, the network will achieve phase desynchronization.
4.3.4 Desynchronization with Heading Rate Constraint

As shown previously, for a robotic network following the control given in (4.1), the effect of the heading rate constraint to the standard PRF in (2.6) can be modeled as a PRF in the form of (4.5) such that the forward coupling function and the backward coupling function cause the phase update rule in (4.6) to be strictly increasing over the entire interval $[0, 2\pi]$ at each firing instance. Thus, we can present the following main theoretical result.

**Theorem 6.** Let $N$ mobile robots, each with heading $\theta_i$, be connected in an all-to-all network topology with no two robots having equal initial heading. Then, with the phase response function given by (2.6) for $l_1 \in [0, 1)$ and $l_2 \in [0, 1)$, the heading control given in (4.1) will desynchronize the headings of the robots with the heading rate constraint given in (4.2).

**Proof.** The proof for this theorem follows directly from the proof for Theorem 5. By using the heading control given in (4.1) such that the initial value for the phase variable $\phi_i$ is set to be $\theta_i(0)$, we have $\theta_i = (\phi_i - \omega_0 t) \mod 2\pi$ for $i \in \mathcal{V}$. Since no two initial headings of the robots are identical, the initial phase variables for each robot will also be non-identical. Thus, to show that the headings of the robots $\theta_1, \ldots, \theta_N$ achieve the state of phase desynchronization, we only need to show that the phase variables $\phi_1, \ldots, \phi_N$ achieve the state of phase desynchronization.

Under the heading rate constraint, the resulting phase response function can be described in the form of (4.5) such that the forward coupling function and the backward coupling function, which will vary with time due to the timing of successive pulses, cause the phase update rule in (4.6) to be strictly increasing over the entire interval $[0, 2\pi)$. Thus, from Theorem 5, the phase variables $\phi_1, \ldots, \phi_N$ will desynchronize following the phase response function given in (2.6) under the heading rate constraint given in (4.2). Therefore, the headings of the robots $\theta_1, \ldots, \theta_N$ will desynchronize.

4.3.5 Remarks on Desynchronization Theorems

If a robot fires a pulse, yet has not completed its desired phase change $\psi_i$ from the previous pulse firing, then the robot must reset its frequency $\omega_i$ back to its fundamental frequency (until the next pulse firing). Otherwise, the resulting phase update rule may not necessarily be increasing over the interval $[0, 2\pi)$, and thus, phase desynchronization is not guaranteed.
If a robot receives a pulse when its phase variable is in the interval $[\frac{2\pi}{N}, \frac{2\pi}{N} - \frac{1}{N}]$, and it has not yet completed its desired phase change $\psi_i$ from the previous pulse firing (when its phase variable was in the interval $(0, \frac{2\pi}{N})$), then the robot may continue to evolve at its current frequency $\omega_i$ determined from the previous pulse firing until it has finished achieving its desired phase change, rather than return to its fundamental frequency. The resulting phase update rule will still be strictly increasing, and the results of Theorem 6 will still apply.

To achieve heading desynchronization, each robot must change its heading at the same rate, $\omega_{\text{max}}$, so that the phase update rule in (4.6) is strictly increasing. If non-identical robots with different maximum rates of rotation are used in the same network, then the network must use some rotation rate that is no larger than the smallest maximum rate.

It is necessary for the robot network to have an all-to-all topology in order to achieve heading desynchronization. Nearly all existing results on pulse-coupled oscillators require this condition to achieve desynchronization among all oscillators. Relaxing this condition necessitates additional limitations to other aspects of the network, such as initial state [2].

It is necessary for no two robots to have identical initial headings in order to achieve heading desynchronization, since they will respond identically to any incoming pulses. To the best of our knowledge, all pulse-based desynchronization algorithms fail in this situation. However, this assumption is not critical, since it is rare for two robots to initialize with the same heading if it is based on analog measurements, such as from a magnetometer. Also, under non-ideal conditions such as pulse propagation delay and pulse drops, it is likely for two robots with identical headings to respond differently to incoming pulses, and thus diverge from each other.

4.4 Experimental Results

We will now evaluate the theoretical results from Sections 4.2 and 4.3 by performing experiments on a physical robotic platform. With these experiments, we can observe the behavior of the robotic network under practical non-ideal conditions, such as pulse propagation delay and pulse drops.
4.4.1 Experimental Setup

We perform these experiments on a group of iRobot® Create 2 programmable robots, or Roombas, as shown in Fig. 4.4. Roombas provide a relatively simple interface with which to control their actions, along with a wide variety of built-in sensors. The movement of a Roomba is controlled using differential steering, allowing for a simple vehicle model on which to experiment.

Each Roomba is outfitted with a Raspberry Pi 3 Model B microcomputer, which is powered by the Roomba’s battery. Programs are written in Python on the Raspberry Pi, allowing the Raspberry Pi to send commands to and receive data back from the Roomba and other sensors. Additionally, a XBee RF communication module is connected to each Raspberry Pi to allow communication between Roombas. To determine the heading of each Roomba, a digital magnetometer can be used to give an initial reading. By using wheel encoder sensor measurements from the Roomba, we track and update the Roomba’s heading over time.
An oscillator is modeled on each Raspberry Pi, where the phase variable is comprised of the sum of the Roomba’s heading and a counter. The counter is based on the internal time of the Raspberry Pi, and increases at the rate $\omega_0$. When a Roomba is not responding to received pulses, the phase variable cycles in the range $[0, 2\pi)$ due to the value of the counter while the Roomba is stationary and its heading is unchanging. At the beginning of each experiment, all counters are reset to zero, such that the initial value of the phase variable is equal to the initial heading of the Roomba. To better replicate and compare the results of the experiments, we initialize the heading of each Roomba to a pre-specified direction.

Since the RF communication module has a very large communication range (approximately 100 meters) compared to the spacing between the Roombas (approximately 10 meters), every Roomba can respond to every other Roomba. To test different network topologies, we can define the connections between Roombas by letting each Roomba respond to pulses from a specific set of Roombas. For a single Roomba, when the phase variable reaches the threshold $2\pi$, a pulse is sent to all connected Roombas through the RF communication module, and the counter value for that Roomba is decreased by $2\pi$ such that the phase variable is reset to zero. When a Roomba receives a pulse through the RF communication module, the current phase (i.e., the sum of the current heading and counter) is used to determine the amount of phase change required by the PCO algorithm, as given in (2.1). This phase change is then applied to the heading of the Roomba. The Roomba spins at a constant rate $\omega_{\max}$ toward the heading at which it should be until the desired heading change is achieved or another pulse is received. Thus, the control of the heading, as given in (4.1) with the necessary heading constraint, is satisfied for each Roomba.

The constraint on the rotation rate cannot be ignored when applying the PCO algorithm to physical systems. Because of the heading rate constraint, the robot may not have completed its desired heading change before a new pulse is received. If it is assumed that the robot has achieved the desired heading change when a new pulse is received, even if it has not, then the robot network will not achieve the desired state. We demonstrate this behavior in Fig. 4.5, showing the evolution of the Roomba headings assuming instantaneous heading adjustment according to the PRC in Fig. 2.2. The headings, rather than synchronizing to the same value, stop adjusting and do not converge. This result confirms the necessity to address heading coordination under rate constraints explicitly.
4.4.2 Heading Synchronization

We now show the experimental results for a group of six Roombas following the synchronization algorithm detailed in Section 4.2.

The Roombas, under an all-to-all topology, begin with initial headings within a containing arc $\Lambda < \pi$. The headings of the Roombas over time are shown in Fig. 4.6(a), in which we see that the headings of the Roombas converge to the same value. As can be seen from the resulting containing arc over time, as shown in Fig. 4.6(b), the network reaches the synchronized state and is stable.

Synchronization of robot headings can also be achieved under more general topologies. We illustrate this point by using the bidirectional ring topology. The headings of the Roombas over time are shown in Fig. 4.7(a), in which we see that the headings of the Roombas converge to the same value. It is apparent from the resulting containing arc shown in Fig. 4.7(b) that the network synchronizes more slowly with the ring topology compared to the all-to-all topology.

Due to the propagation delay of pulses between Roombas, the headings of the Roomba
slowly shift backward over time. This behavior can be reduced by introducing a non-zero refractory period, $D$, such that robots do not respond to pulses received when their individual phase variable is in the region $[0, D)$ [55]. Using the same initial headings for the Roombas, Fig. 4.8(a) shows the evolution of the network with refractory period $D = \pi$. As can be seen from the resulting containing arc evolution in Fig. 4.8(b), the network reaches the synchronized state and is stable.

4.4.3 Heading Desynchronization

We next show the experimental results for a group of six Roombas in an all-to-all topology following the desynchronization algorithm detailed in Section 4.3.
Figure 4.7: (a) Heading evolution of the robots under the heading synchronization algorithm for \( N = 6 \) robots in a bidirectional ring topology, with \( \alpha = 0.5, \ D = 0, \ \omega_0 = \frac{\pi}{2}, \) and \( \omega_{\text{max}} = 0.3\omega_0. \) (b) Containing arc, \( \Lambda, \) as a function of time for the robot headings in (a).

We initialize the Roombas to have headings close together. The evolution of the headings is shown in Fig. 4.9(a), in which we see that the headings of the Roombas diverge, or spread out equally. By observing the desynchronization measure, given in (2.7), as shown in Fig. 4.9(b), the network reaches the desynchronized state, and is stable.

Similar to the synchronization case, the Roomba headings slowly drift backward over time. This behavior is again due to the propagation delay of the pulses, and can be mitigated by reducing the backward coupling strength, \( l_2. \) This mitigation is confirmed by Fig. 4.10(a), which shows the evolution of the robot headings with backward coupling strength \( l_2 = 0. \) The corresponding evolution of the desynchronization measure is shown in Fig. 4.10(b).
Figure 4.8: (a) Heading evolution of the robots under the heading synchronization algorithm for $N = 6$ robots in an all-to-all topology, with $\alpha = 0.5, D = \pi, \omega_0 = \frac{\pi}{5},$ and $\omega_{\text{max}} = 0.3\omega_0$. (b) Containing arc, $\Lambda$, as a function of time for the robot headings in (a).
Figure 4.9: (a) Heading evolution of the robots under the heading desynchronization algorithm for $N = 6$ robots in an all-to-all topology, with $l_1 = 0.8$, $l_2 = 0.6$, $\omega_0 = \frac{\pi}{5}$, and $\omega_{\text{max}} = 0.3\omega_0$. (b) Desynchronization measure, $P$, as a function of time for the robot headings in (a).
Figure 4.10: (a) Heading evolution of the robots under the heading desynchronization algorithm for $N = 6$ robots in an all-to-all topology, with $l_1 = 0.8$, $l_2 = 0$, $\omega_0 = \frac{\pi}{5}$, and $\omega_{\text{max}} = 0.3\omega_0$. (b) Desynchronization measure, $P$, as a function of time for the robot headings in (a).
Chapter 5

Optimal Control Using Reinforcement Learning

In spite of recent advances in the study of pulse-coupled oscillators, there still exist some challenges in achieving the desired state of the oscillator phase values. As seen in Chapters 3 & 4, the phase response function proposed in [55] can only ensure synchronization when the oscillator phases are all within a half-cycle, and when the network experiences ideal environmental conditions. While extensions to this approach show that synchronization can be assured for any initial set of phase values [35, 36, 37], they require that the coupling strength of the network be maintained above a certain threshold, which is not guaranteed under phase continuity constraints.

For any given physical implementation, the initial oscillator phase values may be determined randomly, with no guarantee that the phases are within a half-cycle. Additionally, real networks experience non-ideal environmental factors, such as (random) pulse propagation delay and oscillators with non-identical frequencies, as well as random and potentially dynamic network topologies. Some analysis of synchronizing pulse-coupled oscillator networks under these sorts of non-ideal environmental conditions has been done. However, limitations to these proposed strategies still exist, and direct analysis of the network in some cases is potentially intractable [26, 34, 54, 36, 23].

In oscillator networks, it is desirable to have oscillators achieve synchronization from any initial condition as quickly as possible. The synchronization process can be manipulated by carefully designing the phase response function, which determines how an oscillator should adjust its phase
variable when a pulse is received from another oscillator. An optimal response is one that allows the network to synchronize both quickly and asymptotically regardless of topology or initial distribution of oscillator phases.

Rather than directly analyzing the pulse-coupled oscillator network under real-world, non-ideal environments to find an optimal phase response function, what if we could have the network itself learn what would be the best adjustment to make? Artificial intelligence strategies that utilize prior experience to determine a best future action can help us to design a network that cooperatively learns how to synchronize quickly and effectively.

In this chapter, we propose a reinforcement learning approach to determine an optimal phase response function under both ideal and non-ideal conditions in a PCO network. By simulating a network’s behavior under ideal and non-ideal conditions, we can let the oscillators learn an optimal interaction strategy to improve synchronization probability and speed. Furthermore, this kind of strategy may allow an oscillator to adapt to dynamic network topologies and environmental conditions to ensure synchronization.

5.1 Preliminaries for Non-Ideal Factors

Recall that we are considering a network of $N$ pulse-coupled oscillators, where the quantity $\phi_i \in S^1 = [0, 2\pi)$ is the associated phase of oscillator $i \in \mathcal{V} = \{1, 2, \cdots, N\}$. In this chapter, we will let each oscillator evolves its phase at its own rate, $\omega_i$. Previously, we had each oscillator frequency set to the fundamental frequency, $\omega_0$. However, in the non-identical oscillator frequency case, which may be caused by hardware heterogeneity or differences in the network environment, we will assume each oscillator evolves at a frequency that is from some uniformly random distribution about the fundamental frequency, $\omega_0$. Thus, a period, $T$, of a standard oscillator cycle is $T = \frac{1}{\omega_0}$.

During the normal evolution of the network, when oscillator phase $\phi_i$ reaches the threshold value, $c_\phi = 2\pi$, the oscillator fires a pulse and resets its phase to zero. Previously, neighboring oscillators received this pulse immediately, being notified of the firing instance of an oscillator in the network. Now, we consider the case when neighboring oscillator $j$ receives this pulse after some small (random) amount of time delay, $\tau_{ij}$. This delay in receiving a pulse is primarily due to the physical propagation of the pulse between oscillators, but it may also include the time required for a node to process the incoming pulse.
An oscillator responds to a received pulse by changing its phase $\phi_i$ by $\psi_i$ according to the network coupling strength $l \in (0, 1]$ and phase response function (PRF) $F(\phi)$:

$$\psi_i = lF(\phi_i) = \lim_{\tau \downarrow 0} (\phi_i(t + \tau)) - \phi_i(t) = \phi_i^+ - \phi_i^-$$(5.1)

where the value $l \in (0, 1]$ is the coupling strength of the oscillators in the PCO network and $\phi_i^-$ and $\phi_i^+$ represent the phase of oscillator $i$ immediately before and after receiving a pulse, respectively. Note that we have changed the variable to denote the coupling strength (“$l$” instead of “$\alpha$”) to reduce confusion when discussing the reinforcement learning framework below.

### 5.2 Reinforcement Learning Setup

We want to use reinforcement learning techniques to determine an optimal phase response function $F(\phi)$, as in (5.1), that allows the network to synchronize under both ideal and various non-ideal network factors. Formal, direct analysis for PCO networks under all practical non-ideal network conditions, such as pulse propagation delay, non-identical oscillator frequencies and randomly initialized phase values, proves intractable with current analytic techniques. With reinforcement learning strategies, we can model all the non-ideal factors in the environment, let the oscillators evolve naturally in the network, and gradually determine an optimal response to received pulses. For our purposes, an optimal phase response function $F(\phi)$ is one that ensures that the oscillator both synchronizes its phase with the other oscillators in the network and does so as quickly and efficiently as possible.

To implement reinforcement learning to determine an optimal PRF, we construct a Markov decision process (MDP) [52] with each oscillator in the network as an agent. For each oscillator, the state will be its phase value, $\phi_i$, when a pulse is received, and the action will be the amount the oscillator changes its phase value, $\psi_i$ (under a fixed coupling strength $l$). For simplicity, we will limit the possible actions that an oscillator can take to phase changes that keep it within its current cycle, i.e., for oscillator phase $\phi_i$, $F(\phi_i) \in [-\phi_i, 2\pi - \phi_i]$ holds. The environment for each oscillator consists of the other oscillators in the network along with the network dynamics. Thus, our reinforcement learning strategy consists of a multi-agent system, where the multiple agents form a part of the environment for each individual agent. We desire the agents to cooperate and coordinate...
their actions such that the phase values synchronize.

However, oscillators evolve on the continuous interval $S^1 = [0, 2\pi)$ and, for each phase along that interval, oscillators have a continuous range of possible changes in phase that can be made. Thus, learning the value of each state-action pair is impossible due to the infinite number of state-action pairs. Parameterization and approximation decisions for the state and actions must be made in order to implement reinforcement learning. In the following subsections, we will detail our choices for approximating states and actions, our design decisions for updating state-action values, along with our design decisions for the reward signal.

5.2.1 States, Actions, & State-Action Values

Since the oscillator’s state and action values evolve on continuous intervals, parameterization and approximation decisions must be made to implement reinforcement learning. For our MDP implementation, we parameterize the continuous state interval into $P + 1$ evenly spaced parameters, $s_0, s_1, \ldots, s_P$. The state parameter $s_p$ corresponds to the phase value $\frac{2\pi p}{P}$ in $S^1$.

Additionally, we choose to discretize the actions available for each state parameter with $A + 1$ evenly spaced actions for each state parameter, $a_0, a_1, \ldots, a_A$, such that the actions are limited to phase changes that keep the oscillator within its current cycle. The possible actions that can be taken by an oscillator for state parameter $s_p$ can be expressed as $a_k = -s_p + \frac{2\pi k}{A}$ for $k = 0, 1, \ldots, A$.

We represent the value of each state-action pair with a $(P + 1) \times (A + 1)$ matrix $Q(s, a)$. Each element of $Q(s, a)$ estimates the value of the amount of expected reward by taking the action $a$ when at state $s$.

Remark 8. As the number of state parameters $P$ and discrete actions $A$ increase, the more precisely we can estimate the function $Q(s, a)$, the expected reward for taking an action at a specific state. However, this increase in the number of state-action value pairs will also increases the amount of time required to train the system and estimate these state-action values.

5.2.2 Policy Selection

A policy $\pi$ for our MDP consists of a set of actions, one action for each state parameter $s_p$, such that $\pi(s_p) = a_p$. This policy represents a straight-line approximation of the PRF $F(\phi)$ for a single oscillator.
For our approach, since we are parameterizing the state and discretizing the actions of the oscillator network, we choose to implement episodic reinforcement learning using an on-policy temporal-difference reinforcement technique. Off-policy reinforcement learning techniques, such as Q-learning, tend to perform worse when there is a need to approximate continuous state and action spaces [52].

Since we are choosing to use on-policy learning, we choose a policy to use for each oscillator before each episode. To avoid confusion with the oscillator phase threshold \( c_\phi = 2\pi \), we denote policies with subscripts, such that the policy used for episode \( t \) by oscillator \( i \) is \( \pi_{t,i} \).

The choice for a policy is based on the current state-action value estimates \( Q(s,a) \) for each state parameter. We will use a soft-max, or Boltzmann, distribution to choose an action for the policy at each state parameter. The probability that action \( a_k \) is chosen for the policy at state parameter \( s_p \) is given by

\[
P(a = a_k | s = s_p) = \frac{e^{h(Q(s_p,a_k))}}{\sum_{j=0}^{A} e^{h(Q(s_p,a_j))}}
\]  

(5.2)

where

\[
h(Q(s,a_k)) = \beta \left( Q(s,a_k) - \frac{1}{A} \sum_{i=0}^{A} Q(s,a_i) \right)
\]  

(5.3)

Initially, the values for all state-action pairs are set to zero, i.e. \( Q(s,a) = 0 \) \( \forall s,a \). Thus, the initial policy is equally likely to choose any action for each state parameter. As the state-action values are updated, state-action pairs with larger values become more likely to be chosen, and pairs with smaller values become less likely to be chosen. As the positive parameter \( \beta \) is increased, state-action pairs with larger values become more likely to be chosen. As \( \beta \) increases to infinity, the state-action pair with the maximum value is chosen with probability 1.

5.2.3 Reward Design

Our goal is to have the network’s oscillator phases synchronize. Due to the dynamics of pulse-coupled oscillator networks, the choice of reward and how to update the values of state-action pairs are both critical and challenging. We would like to reward oscillators when the length of the containing arc, \( \Lambda \) in (2.5), decreases, thus causing the network to synchronize, but we also want to penalize oscillators for excessive or unnecessary changes to its phase. Thus, a combination of positive reward for decreasing the length of the containing arc and negative reward for phase adjustments
seems reasonable.

5.2.3.1 Containing Arc Estimate

Since the length of the containing arc of the network measures how well the network is synchronized, we reward actions that decrease the length of the containing arc and penalize actions that increase that length. Thus, the reward will include the decrease in the length of the containing arc, \( \Delta \Lambda = \Lambda^- - \Lambda^+ \), where \( \Lambda^- \) and \( \Lambda^+ \) are the lengths of the containing arc before and after the action was taken, respectively.

The state of the PCO network can be represented as the set of \( N \) phases \( \phi_i \) \( \forall i \in V \). At any time, the oscillator phases completely define the current state of the network, and can be used to calculate the containing arc \( \Lambda \) in (2.5).

However, this global view of the network state cannot be seen by a single oscillator within the network, as the oscillator phases are not directly communicated by the transmission of pulses. Only the difference in phase between one oscillator and another oscillator can be determined, and that only when that second oscillator fires a pulse that is received by the first oscillator. Thus, a local view of the network state can be formed by each oscillator using its own phase and the difference in phase between itself and the other oscillators in the network. For example, when oscillator \( i \) receives the \( m \)th pulse after its own firing (assume this \( m \)th pulse to be from oscillator \( j \)), an estimate of how far ahead the phase of oscillator \( j \) is compared to oscillator \( i \)’s own phase, \( \Delta \phi_{i,m} \), can be determined as

\[
\Delta \phi_{i,m} = (\phi_j^+ - \phi_i^+) \mod 2\pi = 2\pi - \phi_i^+ \tag{5.4}
\]

Since these phase difference estimates are with respect to the phase of oscillator \( i \), an estimate of the phase for each of the other oscillators can be found as \( \hat{\phi}_j = (\phi_i + \Delta \phi_{i,m}) \mod 2\pi \). Therefore, as the network evolves, each oscillator can track the phase of the other oscillators in the network by recording phase difference estimates as in (5.4), and can maintain an independent estimate of the state of the network.

The expression in (5.4) is a stale estimate of each oscillator’s phase by oscillator \( i \). It is “stale” since oscillator \( i \) can only estimate another oscillator’s phase when it receives a pulse from that oscillator [41]. As the network evolves, oscillators will receive additional pulses, and the phase difference between any two oscillators will change. However, oscillator \( i \) will be unaware of these
changes; it is only aware of the phase of an oscillator from which it just received a pulse.

Since each oscillator only knows the phase of another oscillator when it receives a pulse, the oscillator’s local estimate of the network state consists of its own phase, \( \phi_i \), and \( n_i \) phase differences \( \Delta \phi_{i,j} \) for \( j = 1, 2, \ldots, n_i \) as given in (5.4).

Consider oscillator \( i \) that currently has \( n_i \) phase difference values from the previous cycle. When oscillator \( i \) fires a pulse, its phase value resets to zero, and its pulse counter \( m \) resets to zero. When the oscillator receives a pulse (e.g., the first pulse received after firing its own pulse), it adjusts its phase following the phase response function as in (5.1) and increments the pulse counter \( m \) by one. The phase difference estimate for this first oscillator immediately behind oscillator \( i \) in phase, \( \Delta \phi_{i,1} \), is found using (5.4).

For the remaining \( n_i - 1 \) phase differences, the values need to be updated since oscillator \( i \) has adjusted its phase. Since we may not know how these other oscillators may have responded to, or if they even received, the same pulse, it is simplest to assume that these other oscillators have not adjusted their phase value. Thus, we can update each of the remaining phase difference values as

\[
\Delta \phi_{i,j}^+ = (\Delta \phi_{i,j}^- - \psi_i) \mod 2\pi \quad \forall j \neq m
\]  

(5.5a)

where \( \Delta \phi_{i,j}^- \) and \( \Delta \phi_{i,j}^+ \) denote the phase difference value immediately before and after the update respectively.

**Remark 9.** Alternatively, if we can assume that each oscillator is following the same phase response function \( F(\phi_i) \) or policy \( \pi_t \), then we can better estimate the new value for the remaining phase differences. Specifically, the update would be

\[
\Delta \phi_{i,j}^+ = (\Delta \phi_{i,j}^- + \psi_j - \psi_i) \mod 2\pi \quad \forall j \neq m
\]  

(5.5b)

where \( \psi_j = lF((\phi_i^- + \Delta \phi_{i,j}^-) \mod 2\pi) \). For our approach, we only consider the case when the oscillators cannot make this assumption.

This process of updating the phase difference values for oscillator \( i \) continues until its phase reaches a threshold and it fires its own pulse again. The oscillator’s phase and pulse counter reset to zero, and the process repeats. Thus, oscillator \( i \) can estimate the state of the network and the containing arc of the network, and we can use the oscillator’s estimated state to approximate the
change in the containing arc, thus determining the reward value, for a given action.

5.2.3.2 Remarks on Containing Arc Estimate

If, when a pulse is received by oscillator $i$, the counter $m$ increments to a value greater than the number of phase differences it currently has, then the value $\Delta \phi_{i,m}$ is added to the set of phase differences for oscillator $i$’s state estimate. The other phase difference values are updated following the same method as before. This situation can occur if another oscillator passes oscillator $i$ in phase, or when an additional oscillator joins the network, during the current cycle.

If, when a pulse is fired by oscillator $i$, the counter $m$ is smaller than the number of phase differences it currently has, then the values of $\Delta \phi_{i,j}$ for $j > m$ are removed from the set of phase differences for oscillator $i$’s state estimate. This situation can occur if oscillator $i$ passes another oscillator in phase, or when one of the oscillators leaves the network, during the current cycle.

Oscillator $i$ does not know from which specific oscillator a given pulse is received. It keeps track of the number of pulses received after it fires its own pulse, and assigns oscillator $j$ to a certain number of pulses received after it has fired its own pulse.

The specific ordering or reordering of the oscillators in the network has no inherent effect on the phase difference estimates held by oscillator $i$. If two oscillators switch order, and oscillator $i$ receives a different number of pulses between its own pulse firing and oscillator $j$’s pulse, the reordering won’t be known to oscillator $i$. Thus, the reordering is irrelevant. Oscillator $i$ will continue to assign pulses to oscillator $j$ in the prescribed manner.

The estimate of the state of the network by oscillator $i$ will rarely be exactly the same as the true state. The estimate becomes worse as non-ideal factors such as non-identical oscillator frequencies and pulse propagation delay are introduced. However, the stale state estimate never drifts far away from the true state value, so long as each oscillator receives pulses regularly from the other oscillators in the network. Thus, as the true state of the network synchronizes, so will the state estimates of each oscillator.

In the cases involving network topologies other than an all-to-all topology, the state estimate of an oscillator will only include the phases of the oscillators to which it is connected, $\mathcal{N}_i$. Thus, each oscillator will only have partial knowledge of the entire state of the network. But again, as the true state of the network synchronizes, so will the state estimates of each oscillator.
5.2.3.3 Action Penalty Function

Multiple actions can result in the same decrease in the length of the containing arc. To encourage quick and efficient synchronization, we penalize the oscillator based on the magnitude of the action taken. If the oscillator takes an action $a_k$, we will penalize the oscillator by the amount $f(a_k)$, where $f(\cdot)$ is the action penalty function. An action penalty function $f(\cdot)$ should be non-negative on the domain $[-2\pi, 2\pi]$, such that $f(a) \leq |a|$ holds. This requirement ensures that a non-negative amount of reward can be given for state-action pairs.

Many options for an action penalty function exist. A simple choice for the action penalty function is a quadratic function, where $f(a_k) = \frac{a_k^2}{2\pi}$. Another potential choice is a logarithmic function, where $f(a_k) = 2\pi \ln(1 + \frac{|a_k|}{2\pi})$. We will choose the quadratic action penalty function to model the amount of “energy” required by the oscillator to change its phase by $a_k$.

5.2.3.4 Combined Reward Value

Therefore, when an oscillator receives its $k$th pulse and takes an action $a_k$ that decreases the (estimated) length of the containing arc by an amount $\Delta \Lambda_k$, the total reward given to the oscillator for that action will be

$$R_k = w_{\Lambda} \Delta \Lambda_k - w_a f(a_k)$$

(5.6)

where $w_\Lambda$ and $w_a$ are positive weights for the change in the length of the containing arc and the action penalty function respectively.

5.2.4 Episode Structure

During a training episode $t$, we let the network of $N$ oscillators evolve for a fixed amount of time each following a given policy $\pi_{t,i}$. When an oscillator receives a pulse, it uses the current episode’s policy, $\pi_{t,i}$, to determine its action, i.e., phase adjustment, based on a given coupling strength $l$ and its current phase, $\phi_i$.

5.2.4.1 Generating State-Action-Reward Sequences

Let us denote the two closest state parameters to $\phi_i$ for the $k$th received pulse during an episode as $s_{L,k}$ and $s_{H,k}$, respectively, where $s_{L,k} \leq \phi_i \leq s_{H,k}$ holds. The corresponding actions from policy $\pi_{t,i}$ for those state parameters are denoted as $a_{L,k}$ and $a_{H,k}$ respectively. To determine
the action that the oscillator will take, denoted as \( \psi_i \), we weight the actions of the two nearest state parameters based on the proximity of the current phase to those state variables. That is, the phase adjustment \( \psi_i \) in (5.1) is

\[
\psi_i = F(\phi_i) = \rho_{L,k}a_{L,k} + \rho_{H,k}a_{H,k}
\]  

(5.7)

where

\[
\rho_{L,k} = \frac{s_{H,k} - \phi_i}{s_{H,k} - s_{L,k}}
\]  

(5.8a)

and

\[
\rho_{H,k} = \frac{\phi_i - s_{L,k}}{s_{H,k} - s_{L,k}}
\]  

(5.8b)

Note that, for evenly spaced state parameters, the difference between two adjacent state parameters values is \( s_{H,k} - s_{L,k} = \frac{1}{p} \).

Before the oscillator adjusts its phase, it will record its current state, i.e. phase, for the state-action-reward sequence as \( S_k = \phi_i \), where \( k \) is the index for the number of received pulses during an episode. The action taken, based on the policy \( \pi_{t,i} \), is recorded as \( A_k = \psi_i \). The oscillator then calculates its reward, \( R_k \), based on (5.6). The oscillator then evolves freely until another pulse is received. Thus, if a total of \( K \) pulses are received by an oscillator during an episode, the state-action reward sequence for a single oscillator will be as follows.

\[
S_1, A_1, R_1, S_2, A_2, R_2, \ldots, S_{K-1}, A_{K-1}, R_{K-1}, S_K, A_K, R_K
\]

Note that the total number of pulses received by an oscillator may vary for each oscillator in the network per episode due to the choice of the episode policy and resulting network dynamics.

**Remark 10.** During a learning episode, an oscillator only records its state and action when it receives a pulse. It does not record it state and action when it fires a pulse, since we do not consider the resetting of the oscillator’s phase to be an action. Additionally, the oscillator’s estimate of the length of the containing arc does not change when it fires a pulse, since its estimate of the state only changes when it receives a pulse.
5.2.4.2 State-Action Value Update

Once the episode for the oscillator network is complete, we use the resulting state-action-reward sequence for each oscillator to perform a batch update of the state-actions value matrix, \( Q(s,a) \), for that oscillator. The update is based on the Sarsa algorithm [52], where the reward value \( R_k \) will apply to the state-action values for the nearest state parameters to \( S_k \) and their corresponding actions according to the policy \( \pi_{t,i} \).

For the state-action pair \( S_k, A_k \), we update the state-action value for the two state parameters on either side of \( S_k \) and their corresponding actions from the episode policy \( \pi_{t,i} \). The state-action update for the smaller state parameter \( s_{L,k} \) is given by

\[
Q(s_{L,k},a_{L,k}) = Q(s_{L,k},a_{L,k}) + \rho_{L,k} \alpha \left[ R_k + \gamma Q_{E,k+1} - Q(s_{L,k},a_{L,k}) \right]
\]

and the state-action update for the larger state parameter \( s_{H,k} \) is given by

\[
Q(s_{H,k},a_{H,k}) = Q(s_{H,k},a_{H,k}) + \rho_{H,k} \alpha \left[ R_k + \gamma Q_{E,k+1} - Q(s_{H,k},a_{H,k}) \right]
\]

where the coefficient \( \alpha \) is the learning rate, \( \gamma \) is the discount rate, and \( \rho_{L,k} \) and \( \rho_{H,k} \) are calculated using (5.8). \( Q_{E,k+1} \) denotes the average estimated value of the next state-action pair, \( S_{k+1}, A_{k+1} \), based on its nearest state parameters and corresponding policy actions, and can be calculated from the state-action-reward sequence as

\[
Q_{E,k+1} = \rho_{L,k+1} Q(s_{L,k+1},a_{L,k+1}) + \rho_{H,k+1} Q(s_{H,k+1},a_{H,k+1})
\]

This state-action value update is performed for every state-action pair from the episode, except for the final state-action pair from the episode, since there is no corresponding next state-action pair to use in (5.9). We consider this state as the terminal state, and it does not contribute to the state-action value update.

5.2.4.3 Optimal Policy Selection

After all of the state-action value updates have been completed for each oscillator’s state-action-reward sequence, an episode of training is complete. With the updated state-action value
matrix, \( Q(s, a) \), a new policy is chosen for the next episode, and the process is repeated.

After all episodes of training are complete, we can use the values of the state-action matrix \( Q(s, a) \) to determine the optimal policy learned by each oscillator in the network. The optimal policy, \( \pi^*_i \), consists of the actions for each state parameter with the maximum value in the matrix \( Q(s, a) \). That is, \( \pi^*_i(s_p) = \arg \max_{a_k} Q(s_p, a_k) \) holds for each state parameter \( s_p \).

### 5.3 Experimental Results

We now test and verify our proposed reinforcement learning approach and compare our results with previous analytical results by considering networks of different sizes and topologies. In each case, we let the network evolve for \( T = 15 \) seconds for each episode with randomly selected initial oscillator phases in the range \([0, 2\pi)\), and use a network coupling strength of \( l = 1.0 \). We parameterize the state with 101 evenly-spaced parameters and discretize the policy actions into 201 evenly-spaced values, i.e., \( P = 100 \) and \( A = 200 \). For each experiment, we simulate a network for 100,000 episodes. We use a discount rate of \( \gamma = 0.5 \), and have the learning rate \( \alpha \) decrease and the episode policy selection parameter \( \beta \) increase as the network is trained. Additionally, we use \( w_\Lambda = \frac{N}{N-1} \) and \( w_a = \frac{1}{l} \) as the weights in determining the reward value in (5.6).

#### 5.3.1 Ideal Case

We first consider the ideal case, where each oscillator in the network has a frequency equal to the fundamental frequency, \( \omega_0 \), and pulses are received instantaneously with no propagation delay.

We begin our experiments with a simple PCO network of \( N = 2 \) oscillators. Note that this simple network always has a containing arc that is less than one half of the cycle. Fig. 5.1 shows the average of the learned optimal policy, \( \pi^* \), over 10 experiments. This learned phase response closely approximates the standard delay-advance phase response function in [54] obtained under the assumption that the containing arc is less than half of a cycle.

We next consider a network of \( N = 6 \) oscillators in a completely connect, or all-to-all, topology. The average of the learned optimal policy, \( \pi^* \), over 18 experiments, which is shown in Fig. 5.2, is dissimilar to the analytical result in [54], primarily due to the unrestricted initial phase distributions. As the network trains, the randomness in the policy selection does not guarantee that

\[ \text{The Matlab code that we used to generate the results and comparisons in the following sections can be viewed on Github at https://github.com/mathman93/PCO-RL-2022.} \]
the oscillators stay within a half-cycle, even if the phase values were initialized within a half-cycle. Thus, the assumptions for the analytical result in [54] do not hold in this case.

The learned policies from both Fig. 5.1 and Fig. 5.2 can be modeled using the simple form given in (5.11). The best fits to the learned policies using (5.11) are shown in Fig. 5.1 and Fig. 5.2.

$$F_{RL}(\phi) = \begin{cases} -\phi & \text{if } 0 \leq \phi \leq c_1 \\ \frac{\phi}{2\pi - c_1}(\phi - 2\pi) & \text{if } c_1 < \phi \leq c_2 \\ 2\pi - \phi & \text{if } c_2 < \phi \leq 2\pi \end{cases}$$ \hspace{1cm} (5.11)

This PRF model offers important insight into the design principles of the optimal phase response policy. When its phase value is close to the start or end of a cycle, the oscillator learns to take the maximum phase adjustment toward the threshold value, which has been shown analytically to decrease the synchronization time [54]. But when the phase value is near the middle of the
cycle, the oscillator adjusts its phase proportional to the distance to the end of the cycle, similar to the strategy used in [26], which has been shown to almost always lead to synchronization. This combination of two different strategies gives a phase response function that achieves synchronization both quickly and consistently.

We now consider a network of $N = 6$ oscillators in a bidirectional ring topology. Fig. 5.3 shows the average learned optimal policy, $\pi^*$, over 18 experiments along with a best fit using our function model in (5.11). While this learned response is similar to the optimal phase response function found in [54], it is not exactly the same, and is better modeled using (5.11).

Based on our extensive simulations, the learned phase response function $\pi^*$ for a given network seems to be based solely on the indegree of the network, $\delta^-$, rather than the total number of oscillators, $N$. Oscillator with the same indegree in networks of varying sizes tend to have similar learned function parameters, as shown by comparing Fig. 5.4, which shows the average learned
Figure 5.3: Average optimal policy learned using a network of six oscillators in a bidirectional ring topology with identical frequencies and zero propagation delay. The dotted lines show the maximum variations in the learned policies. While the learned phase response is similar to that found in [54], it is better approximated using our response model due to the consideration of global initial phase values.

optimal policy, $\pi^*$, over 15 experiments using an all-to-all topology with three oscillators, to Fig. 5.3. This conclusion is reasonable, since the estimated state of an oscillator is based on the number of oscillators connected to it in the network, since connected oscillators are the only one that can send pulses between each other.

With this simple model of the oscillator’s learned behavior, we now use our reinforcement learning strategy to simulate a variety of network sizes and topologies and determine the function parameters $c_1$ and $c_2$ from (5.11) that best fit the training results. Fig. 5.5 shows these best fit values for multiple runs for each network. These function parameters fit well to an exponential trendline of the form given in (5.12), as shown in Fig. 5.5.

$$c_i = (\pi - b_{i,2})e^{-b_{i,1}(\delta - 1)} + b_{i,2}$$  \hspace{1cm} (5.12)
Figure 5.4: Average optimal policy learned using a network of three oscillators in an all-to-all topology with identical frequencies and zero propagation delay. The dotted lines show the maximum variations in the learned policies. The learned phase response is similar to that found in Fig. 5.3, since the oscillators have the same indegree in both network topologies.

Based on these experiments, the function parameters $c_1$ and $c_2$ in (5.11) can be approximated by the expressions $c_1 \approx 0.81\pi e^{-0.29(\delta^{-1})} + 0.19\pi$ and $c_2 \approx -0.51\pi e^{-0.14(\delta^{-1})} + 1.51\pi$ respectively. These results support our conclusion that the optimal phase response function can be predicted based solely on the oscillator’s indegree.

5.3.2 Non-Ideal Case

We next simulate oscillator networks under non-ideal environmental conditions, including non-identical oscillator frequencies and non-zero propagation delay, and apply our reinforcement learning strategy to determine an optimal PRF. Because the network dynamics are part of the environment of the reinforcement learning process, we can simulate the non-ideal network dynamics and allow the oscillators to learn in the same way as before. For these experiments, we use a network
Figure 5.5: Average values of the learned function parameters in (5.11) using random networks of various sizes. Error bars show the minimum and maximum values learned for a given indegree. Oscillators with the same indegree have similar learned PRFs of the form given by (5.11). A best-fit trendline for the average values of the form given by (5.12) shows that the optimal PRF can be predicted solely based on the indegree of the node.

of $N = 6$ oscillators in an all-to-all topology.

### 5.3.2.1 Non-Identical Oscillator Frequencies

We begin with simulating an oscillator network with non-identical oscillator frequencies. First, we choose an oscillator frequency $\omega_i$ for each oscillator uniformly distributed in the range $[1.98\pi, 2.02\pi]$, i.e., a $\pm1\%$ deviation from the fundamental frequency, $\omega_0 = 2\pi$. The oscillators evolve at that frequency for the duration of the episode, and new frequencies are selected for each episode.

Fig. 5.6 shows the average optimal policy over 18 experiments. Our learned phase response $\pi^*$ is not the same as that found analytically in [54] for the case of non-identical oscillator frequencies.
Figure 5.6: Average optimal policy learned using a network of six oscillators in an all-to-all topology with non-identical frequencies and zero propagation delay. The phase response function was determined with oscillator frequencies uniformly chosen from the range $[1.98\pi, 2.02\pi]$. The dotted lines show the maximum variations in the learned policies. The learned PRF is similar to that found under ideal conditions.

Rather than being a sinusoidal curve, we get nearly the same phase response function as from the previous ideal, identical oscillator frequency case. However, the approach used in [54] adjusts the oscillator frequencies throughout the evolution of the network until they converge to a common value. For the oscillator networks in this paper, we do not adjust the oscillator’s frequency throughout each episode, but only adjust the phase when a pulse is received. Thus, if the oscillator frequencies are not adjusted during the network’s evolution, we conclude that it is optimal for the network to use the phase response function from the ideal case to maintain synchronization.

Increasing the deviation in the oscillator frequencies achieves similar results. Fig. 5.7 shows the average optimal policy over 18 experiments. By choosing the oscillator frequencies $\omega_i$ for each oscillator uniformly distributed in the range $[1.90\pi, 2.10\pi]$, i.e., a $\pm 5\%$ deviation from the fundamental frequency, we again get an learned phase response function similar to the ideal, identical
frequency case.

5.3.2.2 Non-Zero Propagation Delay

We now return to using identical oscillator frequencies, and move to the case of non-zero propagation delay. That is, when oscillator $i$ fires a pulse, oscillator $j$ will not receive the pulse until amount of time $\tau_{ij}$ has elapsed. For our first experiment, we set the propagation delay for each oscillator pair to be a constant $\tau_{ij} = 0.1$ seconds, or one-tenth of an oscillator cycle.

The average optimal policy over 18 experiments is given in Fig. 5.8. This approximated phase response function is again similar to the policy learned in the ideal case. Thus, we conclude that it must be optimal for the network to use the same phase response function as in the ideal case to achieve synchronization, even under fairly extreme propagation delay.
Figure 5.8: Average optimal policy learned using a network of six oscillators in an all-to-all topology with identical oscillator frequencies. The phase response function was determined with constant pulse propagation delay $\tau_{ij} = 0.1 \forall i,j$. The dotted lines show the maximum variations in the learned policies. The learned PRF is similar to that found under ideal conditions.

We now let the oscillators have a uniformly random amount of propagation delay. That is, for the entirety of a single episode, the propagation delay between any two oscillators is uniformly chosen from the range $[0, 0.1]$ seconds for each pulse. The average optimal policy over 18 experiments is given in Fig. 5.9. Again, we see the network learn a phase response policy similar to that in the ideal case.

5.3.2.3 Combined Non-ideal Factors

We finally simulate a network with both random non-identical frequencies and random propagation delay. Since each non-ideal factor tends to not alter the learned phase response policy of the network, we expect that the combination of non-ideal factors will give a similar phase response function as in the ideal case. For this experiment, we choose an oscillator frequency $\omega_i$ for each oscillator uniformly distributed in the range $[1.90\pi, 2.10\pi]$, i.e., a $\pm 5\%$ deviation from the fundamental
frequency, and choose a uniformly random amount of propagation delay from the range $\tau_{ij} \in [0, 0.1]$ seconds for each pulse.

The average optimal policy over 18 experiments is given in Fig. 5.10. As we expect, the learned phase response policy is again similar to that found in the ideal case. Thus, we conclude that our learned phase response function is optimal for the network to achieve synchronization when the network is subject to non-ideal environmental factors.

5.4 Comparison to Existing Results

Given the results of our reinforcement learning strategy for determining an optimal phase response function, we would like to compare this learned PRF to existing analytical results [26, 34, 54, 23] under non-ideal conditions. For our tests, given a network size and topology, we choose an
Figure 5.10: Average optimal policy learned using a network of six oscillators in an all-to-all topology with non-identical frequencies and random propagation delay. The phase response function was determined with oscillator frequencies uniformly chosen from the range \([1.90\pi, 2.10\pi]\) and uniformly random pulse propagation delay \(\tau_{ij} \in [0, 0.1]\) for each episode. The dotted lines show the maximum variations in the learned policies. The learned PRF is similar to that found under ideal conditions.

oscillator frequency \(\omega_i\) for each oscillator randomly distributed about some uniform range centered at the fundamental frequency, and have the pulse propagation delay \(\tau_{ij}\) be randomly distributed in some range for each pulse. We then randomly assign oscillator phase values such that the initial containing arc is greater than half of the cycle threshold, \(2\pi\), and let the network evolve. We can then record whether the oscillator network is able to synchronize and calculate the value of the containing arc once the network has reached steady-state.

The synchronization strategies by Mirollo and Strogatz [26], Nishimura and Friedman [34], and Klinglmayr et.al. [23] do not include a coupling strength parameter. Thus, we modify these algorithms to scale the change in phase adjustment by the coupling strength \(l\), as in (5.1). The original algorithms are recovered under \(l = 1.0\).

We will first consider a small network of \(N = 6\) oscillators in a completely connected
Figure 5.11: The average of the steady-state value of the containing arc for a network of six oscillators in an all-to-all topology with non-identical oscillator frequencies in the range \([1.99\pi, 2.01\pi]\) and uniformly random pulse propagation delay in the range \(\tau_{ij} \in [0.02, 0.04]\) for 2,000 runs. Our learned PRF achieves a greater amount of synchronization for small coupling strength values.

topology. We choose an oscillator frequency \(\omega_i\) for each oscillator uniformly distributed in the range \([1.99\pi, 2.01\pi]\), i.e., a \(\pm 0.5\%\) deviation from the fundamental frequency, \(\omega_0 = 2\pi\), and choose a delay for pulses between any two oscillators to be uniformly random in the range \(\tau_{ij} \in [0.02, 0.04]\).

Fig. 5.11 shows the average value of the containing arc after 2,000 simulations over a broad range of coupling strength values. Our learned PRF is able to synchronize more effectively than the other synchronization strategies at lower coupling strength values. Additionally, we find that our learned PRF is able to synchronize the network more often as the coupling strength decreases, as shown in Fig. 5.12.

We get similar results when we vary the amount of variation in the oscillator frequency and pulse propagation delay. In Fig. 5.13, we show the average value of the containing arc for 2,000 simulations over a broad range of coupling strength values for the same network with an oscillator
frequency $\omega_i$ for each oscillator uniformly chosen in the range $[1.9\pi, 2.1\pi]$, i.e., a ±5% deviation from the fundamental frequency, $\omega_0 = 2\pi$, and propagation delays between any two oscillators chosen uniformly randomly in the range $\tau_{ij} \in [0, 0.04]$. We see that our learned PRF achieves a similar amount of synchronization as the strategies by Wang-Doyle [54] and Nishimura-Friedman [34]. However, as shown in Fig. 5.14, our learned PRF is able to synchronize the network more frequently as the coupling strength is decreased.

Now, we consider how well a ring topology of eight oscillators can synchronize with a variation in oscillator frequency of ±1% (i.e., oscillator frequencies chosen uniformly randomly from the range $[1.98\pi, 2.02\pi]$) and a uniformly random pulse delay of $\tau_{ij} \in [0.01, 0.02]$ seconds. The average value of the containing arc after 2,000 simulations is shown in Fig. 5.15. With small delays, the Mirollo-Strogatz algorithm performs well, achieving good synchronization performance for moderate
coupling strength values, as shown in Fig. 5.16. However, our learned PRF outperforms in both higher and lower coupling strength value ranges.

We also consider eleven oscillators in a “Near4” topology, as shown in Fig. 5.17. We show in Fig. 5.18 how well this topology can synchronize with the same variation in oscillator frequency (±1%) and a broader range of random pulse delay (τ\textsubscript{ij} ∈ [0.01, 0.10] seconds) with the average value of the containing arc after 2,000 simulations. With these larger delays, our learned PRF achieves the greatest amount of synchronization compared to the other strategies over the largest range of coupling strength values. Additionally, our learned PRF synchronizes more frequently as the coupling strength decreases, as shown in Fig. 5.19.

Finally, we consider an Erdős–Rényi-Gilbert (ERG) random topology of N = 10 oscillators, as shown in Fig. 5.20. This topology is not symmetric with respect to oscillator indegree, as were the previously considered topologies. We select the oscillator frequency ω\textsubscript{i} to be uniformly distributed...
Learned PRF
Klinglmayr et.al. (2017)
Wang-Doyle (2012)
Nishimura-Friedman (2011)
Mirollo-Strogatz (1990)

Figure 5.14: The probability of synchronizing for a network of six oscillators in an all-to-all topology with non-identical oscillator frequencies in the range \([1.9\pi, 2.1\pi]\) and uniformly random pulse propagation delay in the range \(\tau_{ij} \in [0, 0.04]\) for 2,000 runs. The learned PRF is able to achieve synchronization more often than the other strategies, especially for small coupling strengths.

in \([1.9\pi, 2.1\pi]\), i.e., a \(\pm 5\%\) deviation from the fundamental frequency \(\omega_0 = 2\pi\), and select delay \(\tau_{ij}\) uniformly in \([0.01, 0.08]\) cycles.

Figure 5.21 shows the average value of the containing arc at steady-state after 2,000 runs over a broad range of coupling strength values. The probability that the network synchronizes is shown in Fig. 5.22. Again, we see that our learned PRF is able to synchronize more effectively and more often as the coupling strength decreases.

Overall, when compared to algorithms by Wang et.al [54], Mirollo & Strogatz [26], Nishimura [34], and Klinglmayr et.al. [23], we find that our learned PRF is able to achieve synchronization more often and to a greater degree when experiencing the combined non-ideal environmental factors of non-identical oscillator frequencies and non-zero, random propagation delays.
Figure 5.15: The average of the steady-state value of the containing arc for a network of eight oscillators in a ring topology with non-identical oscillator frequencies in the range \([1.98\pi, 2.02\pi]\) and uniformly random pulse propagation delay in the range \(\tau_{ij} \in [0.01, 0.02]\) for 2,000 runs. Our learned PRF achieves a greater amount of synchronization over the broadest range of coupling strength values.
Figure 5.16: The probability of synchronizing for a network of eight oscillators in a ring topology with non-identical oscillator frequencies in the range $[1.98\pi, 2.02\pi]$ and uniformly random pulse propagation delay in the range $\tau_{ij} \in [0.01, 0.02]$ for 2,000 runs. Our learned PRF synchronizes more frequently than the Mirollo-Strogatz algorithm [26] at lower coupling strength values.
Figure 5.17: The oscillator network topology used for Fig. 5.18 and Fig. 5.19. Each oscillator node is connected to its two nearest neighbor on both sides.
Figure 5.18: The average of the steady-state value of the containing arc for a network of eleven oscillators in the “Near4” topology with non-identical oscillator frequencies in the range \([1.98\pi, 2.02\pi]\) and uniformly random pulse propagation delay in the range \(\tau_{ij} \in [0.01, 0.10]\) for 2,000 runs. Our learned PRF achieves the greatest amount of synchronization over the broadest range of coupling strength values.
Figure 5.19: The probability of synchronizing for a network of eleven oscillators in the “Near4” topology with non-identical oscillator frequencies in range $[1.98\pi, 2.02\pi]$ and uniformly random pulse propagation delay $\tau_{ij} \in [0.01, 0.10]$ for 2,000 runs. Our learned PRF is able to synchronize the network more frequently than the other synchronization strategies as the coupling strength is decreased.
Figure 5.20: The Erdős–Rényi-Gilbert (ERG) random network topology used for Fig. 5.21 and Fig. 5.22. The probability that any two oscillators are connected was set to 0.3.
Figure 5.21: Average of the steady-state containing arc for ten oscillators in a (ERG) random topology with non-identical oscillator frequencies and random coupling delays. Each run started with randomly selected phases in $S^1$ with a containing arc greater than half of the cycle length.
Figure 5.22: Probability of synchronizing for ten oscillators in a (ERG) random topology with non-identical frequencies and random delay. Each run started with randomly selected phases in $S^1$ with a containing arc greater than half of the cycle length. The learned PRF is able to achieve synchronization more often than the other strategies, especially for smaller coupling strengths.
Chapter 6

Conclusion

In this dissertation, we have considered the challenge of applying the pulse-coupled oscillator network model to physical systems. Prior analysis of PCO networks assumes that the phase variable of the oscillator changes instantaneously when a pulse is received, but this assumption is undesirable, and potentially problematic, if the phase variable is directly associated with a physical quantity. To overcome this challenge, we present a generalization of the standard PCO network model in which the phase variable of the oscillators must evolve in a continuous manner. Our approach avoids the discretization of a continuous control strategy by integrating the discrete-time communication and continuous-time control into a hybrid framework.

Using the presented phase continuity methods, we show that the behavior of the network under these restraints can be modeled as a time-varying coupling strength in the network. Specifically, the coupling strength may be effectively reduced at firing instances. We rigorously and mathematically prove that a pulse-coupled oscillator network can both synchronize and desynchronize with a time-varying coupling strength using the proposed decentralized control framework. Additionally, we find that the reduced coupling strength results in a decrease in the speed to synchronization convergence, and that PCO networks under various synchronization algorithms can still achieve desirable behavior using continuous evolutions in the phase variable.

Due to the inherent non-ideal factors of physical systems, finding an optimal and effective phase response to received pulses is important and challenging. Instead of using a direct analytical approach, we demonstrate a reinforcement learning approach to determine an optimal phase response function for pulse-coupled oscillator networks in non-ideal environments. With each oscillator acting
as an agent of a Markov decision process, cooperative learning of the optimal response to received pulses from the network is achieved. Our approach results in a new set of phase response functions based on the network topology that achieve a greater degree and probability of synchronization when compared to existing strategies.

6.1 Additional Conclusions from Chapter 3

The Peskin and RFA models for pulse-coupled oscillators do not include a coupling strength parameter. Further research may be desirable to consider the effects of having a reduced, and possibly time-varying, coupling strength in the Peskin and RFA models.

Other phase continuity modifications besides the one proposed in this paper are possible. For example, oscillator frequency adjustments can be aggregated if multiple pulses are heard before the desired phase change is complete, resulting in a combined response to multiple firings, rather than a response based on the most recent pulse. Such a design will require separate analysis to ensure the desired convergence properties. General analysis of the myriad of other PCO algorithms under phase continuity and time-varying coupling strength reveals much research potential.

6.2 Additional Conclusions from Chapter 4

We analyze the behavior of the PCO network using two pulse-coupled oscillator algorithms under the phase continuity constraint. Both of these algorithms have been used to study the behavior of oscillators with non-identical frequencies [54, 15]. Future research is required to study the behavior of oscillators having non-identical frequencies and physical networks with heterogeneous continuity constraints.

Additional synchronization and desynchronization algorithms for pulse-coupled oscillators have also been proposed [57, 21, 41, 39, 2]. Further research is needed to determine if these other algorithms can achieve the desired network behavior under the phase continuity constraint and proposed control strategy.
6.3 Additional Conclusions from Chapter 5

Many design decisions were made in forming the reinforcement learning framework to determine an optimal phase response function in non-ideal environments. Further optimizations to the reinforcement learning framework, such as how the reward values and state-action value updates are determined, along with considering adjustments to both oscillator phase and frequency, may lead to additional improvements in determining an optimal phase response function to achieve synchronization. Similar strategies may even be useful in determining an optimal phase response to desynchronize pulse-coupled oscillator networks.
Appendices
Appendix A  Proofs of Lemmas

Statement of Lemma 1

For a network of \( N \) robots with no two robots having equal initial heading, and thus equal initial phase variables, the network will achieve phase desynchronization if the phase response function \( F(\phi_i) \) is given by (4.5) such that the forward coupling function, \( L_1(\phi_i) \in [0, 1) \), over the domain \( (0, \frac{2\pi}{N}) \) and the backward coupling function, \( L_2(\phi_i) \in [0, 1) \), over the domain \( (2\pi - \frac{2\pi}{N}, 2\pi) \) cause the phase update rule in (4.6) to be strictly monotonically increasing.

Proof of Lemma 1

Proof. The proof of Lemma 1 follows the approach given in [15], which gives a proof for a specific case, when the forward and backward coupling strengths are constant over their respective domains. Thus, we need to show that the desynchronization measure in (2.7), \( P \), decreases during each cycle until phase desynchronization is achieved.

To analyze the change of \( P \) at each firing instance, we will utilize the definitions of “active pulse” and “silent pulse” used in [15].

Definition 1. A pulse is an “active pulse” when at least one robot has phase variable \( \phi_i \in (0, \frac{2\pi}{N}) \cup (2\pi - \frac{2\pi}{N}, 2\pi) \) when the pulse is emitted.

Definition 2. A pulse is a “silent pulse” when no robots have phase variable \( \phi_i \in (0, \frac{2\pi}{N}) \cup (2\pi - \frac{2\pi}{N}, 2\pi) \) when the pulse is emitted.

According to Definitions 1 and 2, a pulse is either an “active pulse” or a “silent pulse”. For a “silent pulse”, no robot phase variables are adjusted, according to (4.5). Thus, a “silent pulse” has no effect on phase differences in (2.4) and the measure \( P \). Alternatively, an “active pulse” will cause robot phase variables to adjust, and thus may change the value of the measure \( P \). We can verify the existence of an “active pulse” in each cycle of pulse firings before phase desynchronization is achieved using the results found in [15]. Thus, we need to show that the desynchronization measure \( P \) will decrease at each “active pulse” and converge to zero.

Without loss of generality, let us order the robots by their phase variables such that, at time \( t = 0 \), \( 0 \leq \phi_N < \phi_{N-1} < \cdots < \phi_2 < \phi_1 < 2\pi \) holds. Since the phase update rule is strictly monotonically increasing, then the firing order of robots is invariant, as shown by Lemma 1 in [15]. Thus, if robot \( i \) is the next to fire after robot \( i - 1 \) at time \( t = 0 \), then it will always be the next
to fire after robot $i - 1$. From (2.4) for the length of the arc between two phase variables, we can express that length more simply as:

$$
\begin{align}
\Delta_k &= (\phi_k - \phi_{k+1}) \mod 2\pi, \quad k = 1, 2, \ldots, N - 1 \\
\Delta_N &= (\phi_N - \phi_1) \mod 2\pi
\end{align}
$$

(1)

Note that $\Delta_i = v_i(\phi)$ for all $i \in \mathcal{V}$ holds, and that the desynchronization measure in (2.7) can be equivalently expressed as $P = \sum_{i=1}^{N} |\Delta_i - \frac{2\pi}{N}|$.

Again, without loss of generality, we assume that robot $k$ emits an “active pulse” at time $t = t_k$. From Definition 1, there must be at least one robot with a phase within $(0, \frac{2\pi}{N}) \cup (2\pi \frac{N-1}{N}, 2\pi)$ at time $t = t_k$. Without loss of generality, we assume that there are $M$ robots with phase within $(0, \frac{2\pi}{N})$ and $S$ robots with phase within $(2\pi \frac{N-1}{N}, 2\pi)$, where $M$ and $S$ are positive integers satisfying $2 \leq M + S \leq N - 1$. The $M$ and $S$ phase variables are represented as $\hat{\phi}_{k-M}^-, \ldots, \hat{\phi}_{k-M}^+$ and $\hat{\phi}_{k+S}^-, \ldots, \hat{\phi}_{k+S}^+$ respectively, where the superscript “$\hat{\cdot}$” represent modulo operation on $N$, i.e., $\hat{\cdot} \equiv (\cdot) \mod N$. (Note: 0 maps to $N$.) As a result of our assumptions, we have $\hat{\phi}_{k-M}^- < \frac{2\pi}{N} \leq \hat{\phi}_{k-M}^-$ and $\hat{\phi}_{k+S}^+ < 2\pi \frac{N-1}{N} \leq \hat{\phi}_{k+S}^+$.

Since $\hat{\phi}_{k-M}^-, \ldots, \hat{\phi}_{k-M}^+$ and $\hat{\phi}_{k+S}^-, \ldots, \hat{\phi}_{k+S}^+$ reside in $(0, \frac{2\pi}{N}) \cup (2\pi \frac{N-1}{N}, 2\pi)$, those robots will update their phase variables after receiving a pulse from robot $k$ according to the phase update rule in (4.6), such that

$$
\hat{\phi}_{k-i}^+ = (1 - L_1(\hat{\phi}_{k-i}^-))\hat{\phi}_{k-i}^- + L_1(\hat{\phi}_{k-i}^-)\frac{2\pi}{N}
$$

(2a)

for $i = 1, \ldots, M$, and

$$
\hat{\phi}_{k+j}^+ = (1 - L_2(\hat{\phi}_{k+j}^-))\hat{\phi}_{k+j}^- + L_2(\hat{\phi}_{k+j}^-)\frac{2\pi(N-1)}{N}
$$

(2b)

for $j = 1, \ldots, S$. Note that we also have $\hat{\phi}_{k+q}^+ = \hat{\phi}_{k+q}^-$ for $q = S + 1, \ldots, N - M - 1$ from (4.6), and $\hat{\phi}_{k}^+ = 0$.

According to (1) and (2), the resulting phase variable differences after the phase update, $\Delta_i^+$, caused by an “active pulse” from robot $k$ can be given in seven parts by

$$
\Delta_{k-M-1}^+ = \hat{\phi}_{k-M-1}^+ - \hat{\phi}_{k-M}^- = \hat{\phi}_{k-M-1}^- - (1 - L_1(\hat{\phi}_{k-M}^-))\hat{\phi}_{k-M}^- - L_1(\hat{\phi}_{k-M}^-)\frac{2\pi}{N}
$$

(3a)
\[ \Delta^+_{k-i} = \phi^+_{k-i} - \phi^+_{k-i+1} \]

\[ = [(1 - L_1(\phi^-_{k-i}))\phi^-_{k-i} - (1 - L_1(\phi^-_{k-i+1}))\phi^-_{k-i+1}] + (L_1(\phi^-_{k-i}) - L_1(\phi^-_{k-i+1})) \frac{2\pi}{N} \] (3b)

for \( i = 2, \ldots, M, \)

\[ \Delta^+_{k-1} = \phi^+_{k-1} - \phi^+_{k} = (1 - L_1(\phi^-_{k-1}))\phi^-_{k-1} + L_1(\phi^-_{k-1}) \frac{2\pi}{N} \] (3c)

\[ \Delta^+_k = \phi^+_k - \phi^+_k + 2\pi = 2\pi - (1 - L_2(\phi^-_{k}))\phi^-_{k} - L_2(\phi^-_{k}) \frac{2\pi N - 1}{N} \] (3d)

\[ \Delta^+_{k+1} = \phi^+_{k+1} - \phi^+_{k+1} \]

\[ = [(1 - L_2(\phi^-_{k+1}))\phi^-_{k+1} - (1 - L_2(\phi^-_{k+1+1}))\phi^-_{k+1+1}] + (L_2(\phi^-_{k+1}) - L_2(\phi^-_{k+1+1})) \frac{2\pi N - 1}{N} \] (3e)

for \( j = 1, \ldots, S - 1, \)

\[ \Delta^+_{k+S} = \phi^+_{k+S} - \phi^+_{k+S+1} = (1 - L_2(\phi^-_{k+S}))\phi^-_{k+S} + L_2(\phi^-_{k+S}) \frac{2\pi N - 1}{N} - \phi^-_{k+S+1} \] (3f)

and

\[ \Delta^+_{k+q} = \phi^+_{k+q} - \phi^+_{k+q+1} = \phi^-_{k+q} - \phi^-_{k+q+1} = \Delta^-_{k+q} \] (3g)

for \( q = S + 1, \ldots, N - M - 2. \) Note that (3) can be simplified to that found in [15] if the forward and backward coupling functions are constant over their respective domains.

The new value for \( P \) (denoted as \( P^+ \)) after the update is given by:

\[ P^+ = \sum_{k=1}^{N} |\Delta^+_k - \frac{2\pi}{N}| \] (4)

To determine the change in the value of measure \( P \) caused by the “active pulse” from robot \( k, \) we can calculate the difference between the value of \( P \) before and after the phase update:

\[ P^+ - P = \sum_{k=1}^{N} \left[ |\Delta^+_k - \frac{2\pi}{N}| - |\Delta^-_k - \frac{2\pi}{N}| \right] \] (5)
We can divide (5) into seven parts, using the expressions found in (1) and (3):

\[ P^+ - P = \sum_{k=1}^{N} \left[ |\Delta_k^+ - \frac{2\pi}{N}| - |\Delta_k - \frac{2\pi}{N}| \right] \]

\[ = |\Delta_{k-M-1}^+ - \frac{2\pi}{N}| - |\Delta_{k-M-1} - \frac{2\pi}{N}| \]  
\(\text{(6a)}\)

\[ + \sum_{i=2}^{M} \left[ |\Delta_{k-i}^+ - \frac{2\pi}{N}| - |\Delta_{k-i} - \frac{2\pi}{N}| \right] \]  
\(\text{(6b)}\)

\[ + |\Delta_{k-1}^+ - \frac{2\pi}{N}| - |\Delta_{k-1} - \frac{2\pi}{N}| \]  
\(\text{(6c)}\)

\[ + |\Delta_{k}^+ - \frac{2\pi}{N}| \]  
\(\text{(6d)}\)

\[ + \sum_{j=1}^{S-1} \left[ |\Delta_{k+j}^+ - \frac{2\pi}{N}| - |\Delta_{k+j} - \frac{2\pi}{N}| \right] \]  
\(\text{(6e)}\)

\[ + |\Delta_{k+S}^+ - \frac{2\pi}{N}| - |\Delta_{k+S} - \frac{2\pi}{N}| \]  
\(\text{(6f)}\)

\[ + \sum_{q=S+1}^{N-M-2} \left[ |\Delta_{k+q}^+ - \frac{2\pi}{N}| - |\Delta_{k+q} - \frac{2\pi}{N}| \right] \]  
\(\text{(6g)}\)

The expression in (6b) can be simplified as follows:

\[ \sum_{i=2}^{M} \left[ [(1 - L_1(\phi_{k-i}))\phi_{k-i} - (1 - L_1(\phi_{k-i+1}))\phi_{k-i+1}] \right. \]

\[ + (L_1(\phi_{k-i}) - L_1(\phi_{k-i+1})) \frac{2\pi}{N} \left. - \frac{2\pi}{N} \right] \right] - [(\phi_{k-i} - \phi_{k-i+1}) - \frac{2\pi}{N}] \]

\[ = \sum_{i=2}^{M} \left[ \frac{2\pi}{N} \left( L_1(\phi_{k-i+1}) - L_1(\phi_{k-i}) \right) + L_1(\phi_{k-i})\phi_{k-i} - L_1(\phi_{k-i+1})\phi_{k-i+1} \right] \]

\[ = \frac{2\pi}{N} \left( L_1(\phi_{k-1}) - L_1(\phi_{k-M}) \right) + L_1(\phi_{k-M})\phi_{k-M} - L_1(\phi_{k-1})\phi_{k-1} \]  
\(\text{(7)}\)

where we used the relationships \(\phi_{k-i} - \phi_{k-i+1} < \frac{2\pi}{N}\) and \([(1 - L_1(\phi_{k-i}))(\phi_{k-i} - (1 - L_1(\phi_{k-i+1})))\phi_{k-i+1}\] +
where we used the relationship $\phi_{k-1} < \frac{2\pi}{N}$.

The expression in (6d) can be simplified as follows:

$$|2\pi - (1 - L_2(\phi_{k+1})) \phi_{k+1} - L_2(\phi_{k+1})| \frac{2\pi N - 1}{N} - \frac{2\pi}{N} \left| (2\pi - \phi_{k+1}) - \frac{2\pi}{N} \right|$$

where we used the relationship $2\pi - \phi_{k+1} < \frac{2\pi}{N}$.

The expression in (6e) can be simplified as follows:

$$\sum_{j=1}^{S-1} \left[ \left| (1 - L_2(\phi_{k+j}) - (1 - L_2(\phi_{k+j+1})) \phi_{k+j+1} \right| 
+ (L_2(\phi_{k+j}) - L_2(\phi_{k+j+1})) \frac{2\pi N - 1}{N} - \frac{2\pi}{N} \left| (\phi_{k+j} - \phi_{k+j+1}) - \frac{2\pi}{N} \right| \right]$$

$$= \sum_{j=1}^{S-1} \left[ \frac{2\pi N - 1}{N} (L_2(\phi_{k+j+1}) - L_2(\phi_{k+j})) + L_2(\phi_{k+j}) \phi_{k+j} - L_2(\phi_{k+j+1}) \phi_{k+j+1} \right]$$

$$= 2\pi \frac{N - 1}{N} (L_2(\phi_{k+S}) - L_2(\phi_{k+1})) + L_2(\phi_{k+1}) \phi_{k+1} - L_2(\phi_{k+S}) \phi_{k+S}$$

where we used the relationships $\phi_{k+j} - \phi_{k+j+1} < \frac{2\pi}{N}$ and $|1 - L_2(\phi_{k+j})| \phi_{k+j} - (1 - L_2(\phi_{k+j+1})) \phi_{k+j+1} + (L_2(\phi_{k+j}) - L_2(\phi_{k+j+1})) \frac{2\pi N - 1}{N} < \frac{2\pi}{N}$ for $j = 1, \ldots, S - 1$.

The expression in (6f) can be simplified as follows:

$$\sum_{q=S+1}^{N-M-2} \left| \Delta_{k+q}^+ - \frac{2\pi}{N} \right| - \left| \Delta_{k+q}^- - \frac{2\pi}{N} \right|$$

$$= \sum_{q=S+1}^{N-M-2} \left[ \left| \Delta_{k+q}^+ - \frac{2\pi}{N} \right| - \left| \Delta_{k+q}^- - \frac{2\pi}{N} \right| \right] = 0$$

Thus, combining (6a), (6f), and (7)–(11), we can write (5) in two main parts:

$$P^+ - P = \sum_{k=1}^{N} \left[ \left| \Delta_k^+ - \frac{2\pi}{N} \right| - \left| \Delta_k^- - \frac{2\pi}{N} \right| \right]$$
\[
\Delta_k^+ - \frac{2\pi}{N} - |\Delta_{k-M-1} - \frac{2\pi}{N}| + L_1(\phi_k-M)(\phi_k-M - \frac{2\pi}{N}) \tag{12a}
\]
\[
+ |\Delta_{k+S}^+ - \frac{2\pi}{N} - |\Delta_{k+S} - \frac{2\pi}{N}| + L_2(\phi_{k+S})(2\pi N - 1) - \phi_{k+S} \tag{12b}
\]

This result in (12) is similar to that found in [15]. Since \(L_1(\phi_{k-M})\) and \(L_2(\phi_{k+S})\) are constants in the range \([0, 1)\), then a direct comparison can be made with Part A and Part B in [15] to (12a) and (12b), respectively. Borrowing the conclusion made in [15], we conclude that \(P^+ - P \leq 0\) holds and cannot always be zero, and thus the measure \(P\) in (2.7) will decrease to zero.

Therefore, the network will achieve phase desynchronization using a phase response function given by (4.5) such that the forward coupling function, \(L_1(\phi_i) \in [0, 1)\), over the domain \((0, \frac{2\pi}{N})\) and the backward coupling function, \(L_2(\phi_i) \in [0, 1)\), over the domain \((2\pi N - 1, 2\pi)\) cause the phase update rule in (4.6) to be strictly increasing. \(\square\)
Bibliography


