ABSTRACT

Title of Dissertation: DATA-DRIVEN PREDICTION, DESIGN, AND CONTROL OF SYSTEM BEHAVIOR USING STATISTICAL LEARNING

Xiangxue Zhao, Doctor of Philosophy, 2021

Dissertation directed by: Professor Shapour Azarm, Advisor
Professor Balakumar Balachandran, Co-advisor
Department of Mechanical Engineering

The goal in this dissertation is to develop new data-driven techniques for prediction, design, and control of the behavior of a variety of engineering systems. The data used can be obtained from a variety of sources, including from offline, high-fidelity system's simulation, physical experiments, and online, sparse measurements from sensors. Three inter-related research directions are followed in this dissertation. Following the first direction, the author presents a multi-step-ahead prediction technique for evaluating a single-response (or single-output of the) system's behavior through an integration of the data obtained offline from the system's high-fidelity simulation, and online from single sensor measurements. With regard to the first research direction, the key contribution includes a reasonably fast and accurate prediction strategy that can be used, among others, for online, multi-step ahead forecasting of the system's operational behavior. Building on the work from the first direction, the author follows a second research direction to present a multi-step ahead prediction technique, this time for a

multi-response system's behavior, that can be used for evaluating various system's designs and corresponding operations. Data in this case is obtained from the offline, high-fidelity system's simulations, and online sparse measurements from multiple sensors (or limited number of physical experiments). The main contribution for this second direction is in construction of a new data-driven, multi-response prediction framework that has a robust predictive capability. Along the third research direction, a data-driven technique is used for prediction and co-optimization of a system's design and control. The data in this case is obtained from sensor measurements or a simulator. The main contribution achieved through the third direction is a new data-driven reinforcement learning-based prediction and co-optimization approach.

The methods from this dissertation have numerous applications, including those demonstrated here: (i) assessment of safe aircraft flight conditions (Chapters 2 and 3), (ii) evaluation of design and operation of a robotic appendage (Chapter 3), and (iii) design and control of a traffic system (Chapter 4).

DATA-DRIVEN PREDICTION, DESIGN, AND CONTROL OF SYSTEM BEHAVIOR USING STATISTICAL LEARNING

by

Xiangxue Zhao

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park, in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2021

Advisory Committee:
Professor Shapour Azarm, Advisor and Chair, Mechanical Engineering
Professor Balakumar Balachandran, Co-advisor, Mechanical Engineering
Professor Nikhil Chopra, Mechanical Engineering
Professor Mark Fuge, Mechanical Engineering
Professor Jeffrey Herrmann, Mechanical Engineering
Professor Kayo Ide, Dean's Representative, Atmospheric and Oceanic Science

To My Beloved, Olivia

Acknowledgments

I would like to thank my co-advisors, friends, and family for all their help and support throughout my entire Ph.D. study. This dissertation would not have been possible without their unwavering encouragement and support.

First and foremost, I would like to express my sincere gratitude to my advisor and dissertation committee chair, Dr. Shapour Azarm, and my co-advisor Dr. Balakumar Balachandran, for their patient guidance and numerous feedbacks. Both Dr. Azarm and Dr. Balachandran are outstanding professors and scholars. During my dissertation research, they were always there to push me explore new ideas while communicating them clearly. They also helped me develop necessary technical skills and sharpen my academic instincts. Their mentorship has been invaluable for me academically and professionally.

I also want to thank the other members of my dissertation committee, Dr. Nikhil Chopra, Dr. Mark Fuge, Dr. Kayo Ide, and Dr. Jeffrey Herrmann, who despite their very busy schedule graciously agreed to serve on my committee and provide constructive comments and suggestions for improving this dissertation.

I would like to especially acknowledge and thank the following individuals for their help with this dissertation: (i) Dr. Preidikman of the National University of Córdoba in Córdoba, Argentina, and Mr. Abu Kebbie-Anthony for the help in generating the aeroelastic response data used in Chapters 2 and 3, (ii) Dr. Guanjin Wang for helping and providing the needed information for the robotic appendage motion simulator and pointing me to the published experimental data used in Chapter 3, and finally (iii) Dr. Manuel Rodriguez for the help in developing the traffic road network simulator which

was used in Chapter 4. These individuals have provided some critical resources and data for this dissertation, and I have always enjoyed thoughtful discussions with them. Many thanks are also extended to my other peers, Dr. Tianchen Liu and Mr. Randall Kania for many fruitful discussions. Finally, special thanks to my family: Their unconditional love and support have been the key driving force behind my growth and success. I will be always grateful to them.

Some of the results reported in Chapters 2 and 3 were possible, thanks to partial funding support from the U. S. Air Force Office of Scientific Research (AFOSR), the DDDAS Program, through Grant FA95501510134. This support is gratefully acknowledged.

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

This chapter begins with an introduction in Section 1.1. It is then followed in Section 1.2 with a description of three research questions considered in this dissertation. Finally, in Section 1.3, the organization of the dissertation is provided.

1.1 Introduction

Data-driven prediction techniques for dynamical systems can have many applications, including those for the design, operation, and control of such systems. For instance, being able to estimate an aircraft's structural response for certain flight maneuvers can be extremely important for its safe, online operational decisions. So are just-in-time prediction of unsafe deformations along a railway track, or timely prediction of corrosion damage for water, oil, and gas pipelines, and subsequent corrective design, operation, and control decisions.

The author's aim is to develop and provide statistical learning techniques for prediction, design, and control of the behavior of a variety of engineering systems by using and fusing various types of data. It is assumed that system's behavior (hereafter also referred to as system's response or system's output) is dynamical (is a function of time) and can be partially observed by data collected from the following sources: (i) system's simulations (which can be high-fidelity) and (ii) single sensor or sparse multi-sensor measurements or available data from physical experiments data.

The proposed techniques in this dissertation can have many applications. Three such examples are used in this dissertation to demonstrate applicability of these techniques. For the first application, a model for a fixed wing aircraft example is considered which

has long, thin wings to extend the performance capabilities. This aircraft can be susceptible to aeroelastic flutter behavior, which can result in a premature failure. To avoid such a failure, a good prediction of the aircraft's aeroelastic behavior is necessary for determining safe maneuvers during its operation. This example is further detailed in Chapters 2 and 3.

As a second example, the design and operation of a robotic appendage on a terrain with granular material are considered. For the motion of this robotic appendage on the terrain, a predictive model is constructed and used to evaluate its various designs and operational conditions. This example is further detailed in Chapter 3.

As a third example, the traffic behavior (or performance) for a road network is considered. The traffic performance in a road network can be predicted, among others, by computing a measure of the vehicular queue length and/or vehicular throughput at intersections. After predicting the traffic behavior, one can co-optimize the traffic performance by designing the vehicular flow directions and controlling traffic signals. This example is further discussed in Chapter 4.

1.2 Research Questions and Directions

Three research questions are considered. These research questions and corresponding directions are reviewed in the following.

Research Question 1 (RQ1) -This question reads as follows:

Is there a way to predict a single output response (representing behavior) of a system, multiple time steps ahead, by fusing data obtained offline from the system's high-fidelity simulations, and data obtained online from single-sensor measurements?

The results from RQ1 can be used for online forecasting and evaluation behavior of a single-response system. As a result of the investigation of RQ1, a new RECursive

Multi-Input-Multi-Output (REC-MIMO) approach has been developed. This approach can be used to obtain real-time estimation for a system's response by considering a combination of expensive simulation data, which are obtained during an offline phase, and sensor measurement data, which are obtained during an online phase. In the offline phase, a Multi-Input-Multi-Output (MIMO) strategy is employed for constructing a predictive model for the system's dynamical responses from simulation data. The offline phase data is then prepared for use during the online phase through a combination of Proper Orthogonal Decomposition (POD) and Gaussian Process (GP). Then, by utilizing measurement data collected from sensor, a recursive strategy is used to iteratively construct and enhance multi-step-ahead predictions through a Particle Filter (PF) method during the online phase. An example involving aeroelastic responses for a joined-wing SensorCraft is used to demonstrate the effectiveness of the proposed framework. Through this example, it is shown that the system's state predictions obtained through the REC-MIMO strategy have a comparable accuracy to those obtained the from high-fidelity, system's simulations while significantly reducing computational expense. In Chapter 2, the author gives in detail the results of the investigation carried out in response to RQ1.

Research Question 2 (RQ2) – RQ2, which can be considered as an extension of RQ1, is as follows:

How can we integrate data obtained offline from high-fidelity simulations with limited, online multi-sensor measurements (or physical experiments) to (i) predict multiple output responses (representing behavior) of a system, multiple time steps ahead, and (ii) use the prediction to evaluate various system designs and/or operations?

The results from RQ2 can be used for online forecasting and evaluation of the behavior of a multi-response system. As a result of the investigation of RQ2, a new

approach has been developed based on an integration of dimension reduction, surrogate modeling, and data assimilation techniques. A step-by-step application of the proposed approach is demonstrated through a numerical example. The performance of the approach is also evaluated with two complex engineering examples. The first example involves predicting multi-responses aeroelastic behavior of a joined-wing aircraft in which sensors are sparsely placed on its wing. The second example is about estimating and evaluating robotic appendage motion for various designs and operational conditions, in which limited number of high-fidelity system's simulations and physical experiments data obtained from the literature are used. Through two case studies, it is shown that the results obtained from the proposed prediction technique have comparable accuracy to those from the high-fidelity simulation, while at the same time significant reduction in computational effort is achieved. Chapter 3 contains details on the results of the investigation in response to RQ2.

Research Question 3 (RQ3): This question is posed as follows:

How can the prediction of a system's behavior, using data collected from simulations or sensors, be used for simultaneous optimization (or co-optimization) of a system's design and control?

As a result of the investigation of RQ3, a new data-driven approach is used to predict system's behavior, which is then used for co-optimization of the system's design and control. This research question is investigated in the context of traffic system problems by considering design of vehicular flow directions and traffic signal control policy for road networks to reduce traffic congestion. For this problem, among others in the literature, Reinforcement Learning (RL) has been used for deciding on an "optimum" traffic signal control policy to alleviate congestion in a road network. However, the

traffic signal control policy can also be optimized in conjunction with the design of vehicular flow directions to further improve congestion using an extended RL-based technique. The proposed RL-based technique is evaluated by using two examples under rush hour and non-rush hour traffic conditions. It is found that, compared to a conventional RL-based approach in which only a traffic signal control policy is considered, the proposed extension of the RL-based approach can obtain a better traffic performance in terms of vehicular queue length and throughput. In Chapter 4, the author covers in detail the results of the investigation obtained in response to RQ3.

1.3 Organization of the Dissertation

The organization of the rest of this dissertation is as follows. In Chapter 2, the author provides details of an approach for predicting behavior of a single-response system, multiple time steps ahead, by fusing data obtained from the system's high-fidelity simulation and single-sensor measurements. Here, a dynamic data-driven prediction approach is developed, which consists of an integration of dimension reduction, surrogate modeling and data assimilation techniques. The applicability of the proposed prediction approach in this chapter is demonstrated by predicting and then evaluating the goodness of a single aeroelastic output obtained for a joined wing SensorCraft.

In Chapter 3, the prediction approach from Chapter 2 is extended. Specifically, the goal in Chapter 3 is to predict behavior of a multi-response (or multi-output) system, multiple time steps ahead, by fusing data collected from the system's high-fidelity simulations, and data obtained from a limited number of multi-sensor measurements (or physical experiments). The applicability of the proposed multi-response prediction approach in this chapter is then demonstrated with two engineering examples. In the

first example, the goodness of the prediction is assessed by fusing data obtained from a high-fidelity multi-response aeroelastic model for the joined-wing SensorCraft, with the data obtained from a sparse multi-sensor measurement. In the second example, the goodness of the prediction is assessed using an integration of data obtained from a high-fidelity multi-response simulation model for a robot appendage motion, with the data collected from the literature for a limited number of physical experiments. The goodness of the prediction approach is then assessed for evaluating system performance, in the first example by changing the locations of the sparse sensors, and in the second example by changing the design and operational conditions for the robotic appendage.

Next, in Chapter 4, the goal is to use a data-driven approach, which can be used with simulation or sensor data, to jointly optimize the design and control of a system. The approach in Chapter 4 is constructed in the context of a traffic system. Specifically, the prediction of the traffic behavior is used to co-optimize the design of vehicular flow directions and control of traffic signals at intersection for road networks.

Finally, in Chapter 5, a summary of the research results, highlights of the contributions and suggested future directions are provided.

The reading order of the dissertation is as follows. After reading Chapter 1, the reader may follow with Chapter 2 and then Chapter 3. Chapter 4 can be read after Chapter 1. Finally, Chapter 5 can be read for the dissertation's conclusions.

In the next chapter, the results of the investigation associated with research question 1 are presented.

Chapter 2: Data-Driven Multi-step-ahead Prediction: An Application to Aeroelastic Response for SensorCraft

In this chapter, the outcome of the author's research due to Research Question 1 is presented, i.e., *is there a way to predict a single output response* (*representing behavior) of a system, multiple time steps ahead, by fusing data obtained offline from the system's high-fidelity simulations, and data obtained online from single-sensor measurements?*

This chapter is organized based on the author's previous articles[1,2]. Accordingly, first, a nomenclature section is provided which is specifically tailored for the material presented in this chapter. After that the multi-step-ahead prediction problem formulation and assumptions are detailed in Section 2.1. Next, related literature is provided in Section 2.2. Then, a prediction strategy for evaluating system behavior (or response), called REC-MIMO, is presented in Section 2.3. REC-MIMO consists of three steps: principal component analysis, Gaussian process, and particle filter. In Section 2.4, a nonlinear aeroelastic case study is used to demonstrate the effectiveness of the proposed prediction approach. The results obtained from the approach is compared against those obtained from a high-fidelity simulation. Meanwhile, the

[1] Zhao, X., Kebbie-Anthony, A., Azarm, S. and Balachandran, B. (2019) "Dynamic Data-Driven Multi-Step-Ahead Prediction with Simulation Data and Sensor Measurements." *AIAA Journal*, 57(6), pp. 2270-2279.
[2] Zhao, X., Kebbie-Anthony, A., Azarm, S. and Balachandran, B. (2018) "Dynamic Data-Driven Aeroelastic Response Prediction with Discrete Sensor Observations." *In 2018 AIAA Non-Deterministic Approaches Conference*, p. 2173, Kissimmee, FL, January 8-12.

computational cost of the prediction procedure is studied. Finally, a summary of the chapter is given in Section 2.5.

Nomenclature for Chapter 2:

a	total number of multi-step-ahead time steps
d	index for POD basis
D	number of POD bases
$f(\cdot)$	system dynamic function
$g(\cdot)$	sensor measurement function
i	index for i^{th} training data
I	total number of training datasets
k	index for time step
k_0	initial time step
m	dimension of system input
n	index for n^{th} particle
N	total number of particles
N_{eff}	effective sample size
R_k	statistical properties of sensor noise at k^{th} time step
s_k	sensor measurement data at k^{th} time step
$\boldsymbol{U}, \boldsymbol{V}$	orthogonal matrices
v_k	sensor measurement noise at k^{th} time step
V	airspeed
w	weight for particle
$\boldsymbol{x}$	system input for operating conditions
y_k	system output at k^{th} time step

$\boldsymbol{Y}$	available training data
$\boldsymbol{z}$	random variables model for coefficients of POD bases
α_{id}	coefficient of ith training data for d^{th} POD basis
ρ	air density
$\boldsymbol{\Lambda}$	diagonal matrix
ε	noise for perturbed parameter particles
$\boldsymbol{\phi}$	subspace of $\boldsymbol{Y}$
φ_d	d^{th} POD basis
$\delta(\cdot)$	Dirac delta function

Acronyms

AoA	Angle of Attack
DIRMO	Direct Multi-Input-Multi-Output
EKF	Extended Kalman Filter
EnKF	Ensemble Kalman Filter
FE	Finite Element
GP	Gaussian Process
MIMO	Multi-Input-Multi-Output
pdf	probability density function
PF	Particle Filter
POD	Proper Orthogonal Decomposition
PSO	Particle Swarm Optimization
REC-MIMO	Recursive Multi-Input-Multi-Output
UVLM	Unsteady Vortex Lattice Method

2.1 Multi-step-ahead Prediction Problem Formulation and Assumptions

For this problem, the goal is to efficiently estimate the system's multi-step-ahead dynamical behavior (a time steps ahead of the initial time step k_1). In Figure 2.1, a block diagram of the multi-step-ahead forecasting problem for a system's dynamical behavior is depicted. A system's single-response dynamical behavior is represented by the scalar y_k, which is given by the function $y_k = f(\boldsymbol{x}, k)$. Here, $\boldsymbol{x} \in \mathbb{R}^m$ indicates a vector of the system inputs, where m indicates the dimension of the inputs, and k is an index for time in the range $k \in [k_1, k_a]$, where k_1 and a indicate the initial time step and the total number of time steps of the prediction horizon, respectively. The system's dynamical behavior is assumed to be nonlinearly changing along the time scale as well as for different values of system inputs (for different operating conditions), $\boldsymbol{x}$. Since the response y_k can change with respect to time for a system input $\boldsymbol{x}$ (which is multi-input), hereafter such a system is called as a Multi-Input Multi-Output (MIMO) system.

In this problem, it is assumed that a set of data, $\{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1}^{I}$, is available from the offline phase through a high-fidelity simulation model, where I is the total number of datasets, i is the index of each dataset, and $\boldsymbol{y}$ is a vector of time series system responses for the prediction horizon $\boldsymbol{y} = [y_1, y_2, \dots, y_a] \in \mathbb{R}^a$. The dataset, $\{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1}^{I}$, contains the system's dynamical responses for different system inputs. On the other hand, during the online stage, it is assumed that there is a stream of data from a single-sensor measurement, s_k, of the system at each time step, k, which contains a certain amount of noise, v_k. Here, the relationship between sensor measurements and the system behavior, y_k, is given by the function $g(\cdot)$.

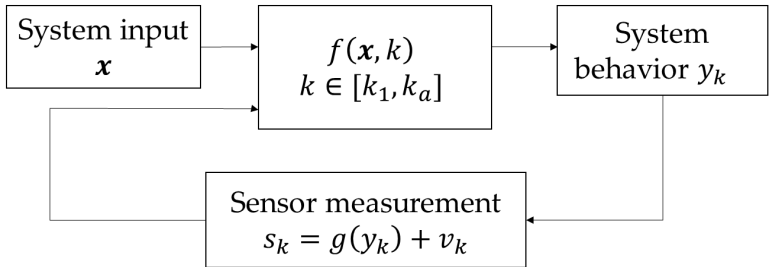


Figure 2.1 Block diagram of the problem.

As shown in the case study later in this chapter, by combining the available high-fidelity simulation datasets (obtained offline) with noisy sensor measurements, the proposed strategy (detailed in Section 2.3) can obtain fast and reasonably accurate long-term prediction for the system's dynamical behavior.

2.2 Related Literature Review

To ensure safe and reliable operation of complex systems in real-time, it is crucial to have a fast and accurate dynamical behavior prediction capability. However, efficient prediction of future behavior of a complex dynamical system remains a challenge. For instance, consider an unmanned aircraft system, such as the joined-wing SensorCraft [1, 2], which is designed with thin and highly flexible wings to enhance mission efficiency. For this aircraft, nonlinear fluid-structure interactions can result in undesired aeroelastic effects, such as flutter, which can result in structural failure. While a high-fidelity nonlinear aeroelastic simulation model can provide insights into structural responses (or behavior) of the aircraft, it is impractical to carry out such simulations online during aircraft operations due to the high computational expense. On the other hand, sensors placed on an aircraft can be used to collect structural response data at various locations of the aircraft during its operation. However, sensor

data alone are not sufficient for prediction of precursors for aeroelastic instabilities such as flutter. Thus, to obtain efficient predictions for a system's online operation, methods for reduction in the computational cost of simulations, as well as preservation of an acceptable level of accuracy, need to be developed. This is the main motivation for Dynamic Data-Driven Application System (DDDAS), under which the current chapter falls. DDDAS has become an overarching framework, whereby application simulation data fused with sensor measurement data can help form a symbiotic feedback control system [3]. In this framework, one can dynamically incorporate measurement data into a running application (e.g., offline high fidelity aeroelastic simulator) for improved accuracy and reduced computational cost. The DDDAS concept has numerous applications in the areas of environmental analysis (e.g., weather forecasting [4], wildland fire detection [5, 6], airborne contaminants identification [7-9]); unmanned vehicle systems (e.g., unmanned aerial vehicle [10, 11], unmanned ground vehicle coordination [12]); and image processing (e.g., target tracking [13]).

As mentioned before, the aim of this chapter is to predict a system's behavior in real-time multiple steps ahead. Unlike one-step-ahead prediction, a multi-step-ahead prediction is much more challenging and can be used to predict the state of the system in two- or several time steps ahead based on past observations, while handling issues such as accumulation of errors and increased uncertainty for a long prediction horizon [14]. The existing modeling strategies in the literature for multi-step-ahead predictions include recursive, direct, Multi-Input-Multi-Output (MIMO), and hybrid techniques. With the recursive strategy [15-18], one constructs a one-step-ahead prediction model, and then utilizes each predicted system response as the input for estimating the response

for the next time step. Examples of the recursive approaches are recurrent neural network [19] and iterative local learning techniques [20]. As opposed to the recursive strategy, in which one uses a single model, with the direct strategy one estimates multi-step-ahead values with multiple forecasting models, wherein from each model one obtains a predicted value for a single time step. Since an iterative model prediction scheme may cause an accumulation of error, it can be concluded from most studies that the direct strategy is better than the recursive approach in terms of prediction accuracy. However, with the direct strategy, one incurs a significantly large computational expense, especially, in the case of a long prediction horizon. The MIMO strategy [21-23] has been proposed to obtain a vector of multiple dynamical responses in a single step. Compared with other strategies, with this MIMO strategy, one can preserve the stochastic dependency amongst the dynamic responses at different time steps. The MIMO strategy has been recognized to outperform other strategies in a comparative study [24].

On the other hand, active research has been ongoing to develop new hybrid strategies based on the abovementioned modeling strategies. For example, the DirRec strategy [25] was introduced to integrate aspects from both the direct and recursive strategies. This means that different models are used to directly estimate at each time step, as the predictions from previous time steps are iteratively fed into the models for later time steps as inputs. The Direct Multi-Input-Multi-Output (DIRMO) strategy [26-29] has been presented to combine the most appealing aspects of the direct and MIMO strategies. In the DIRMO strategy, one trades off the model's capability to characterize the stochastic dependency with the flexibility of the modeling procedure. In reference

[29], further study was performed to extend the DIRMO strategy into Particle Swarm Optimization (PSO). Here, PSO is used to self-adaptively determine the number of sub-models with varying prediction horizons. With the combined DIRMO with PSO approach, there is improvement in model flexibility and accuracy but at the cost of higher computational expense.

In this chapter, a new prediction strategy is proposed, called RECusive Multi-Input-Multi-Output (REC-MIMO), in which the recursive strategy and MIMO strategy are integrated. The proposed approach is designed to provide efficient multi-step-ahead predictions of a single-response system's behavior in real-time. Following the approach, first, the MIMO strategy is employed to build a compact model offline based on the data obtained from a high-fidelity (computationally expensive) dynamical simulation model. This is achieved through Proper Orthogonal Decomposition (POD) [30-32] and Gaussian Process (GP). Here, POD is used to extract the most dominant dynamical features from the available simulation data, and then, GP models are constructed to interpolate the system's behavior. Next, a recursive strategy is used to iteratively improve model predictions with collected noisy sensor measurement data, wherein updating is achieved through a Particle Filter (PF) method [32]. In this way, one can propagate uncertainties through a nonlinear model (simulation) without any assumptions on the distributions of the state variables. However, to dynamically update the multi-step-ahead predictions, the PF might suffer from the "curse of dimensionality", especially, when the prediction time horizon is long. Hence, an extension of PF has been proposed to make it more efficient. These are further detailed in the following section.

2.3 Proposed Prediction Framework

The framework for the proposed REC-MIMO strategy is composed of an offline phase and an online phase, a schematic of which is shown in Figure 2.2. During the offline phase, a MIMO model is employed to directly predict the system's multi-step-ahead response for varying system inputs. This is achieved through a combination of the POD (see Section 2.3.1) and GP (see Section 2.3.2) techniques. First, POD is exploited to extract the most significant features of the prediction horizon (POD bases $\boldsymbol{\phi}$), with each simulation response is approximated by low-dimensional coefficients $\boldsymbol{\alpha}$. Surrogate models are then constructed to interpolate the variation of the coefficients for different system configurations by using GP. Together, those features, and GP models are stored offline for use during the online phase. During the online stage, the GP models are used to make predictions of the coefficients for a given system input. Through the integration of the estimates of the coefficients with POD bases $\boldsymbol{\phi}$, a priori multi-step-ahead prediction can be achieved. Then, with the collected sensor measurements, s_k, at each time step, the prior estimates will be recursively updated through the use of the PF (see Section 2.3.3).

Further details on this framework (REC-MIMO strategy) are discussed in the following subsections.

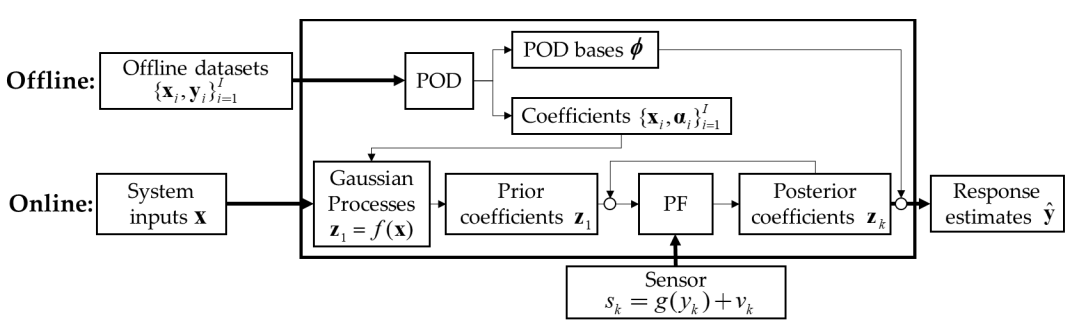


Figure 2.2 Framework for the proposed prediction strategy.

2.3.1 Proper Orthogonal Decomposition (POD)

In the context of dynamical systems [30, 31], POD is usually recognized as a method to construct simplified meta-models for high-dimensional problems to improve computational efficiency. Here, POD is used for extracting a set of significant features from offline simulated data. These responses are represented with a substantially lower dimensional set of parameters, while preserving dominant characteristics of the original simulation data.

Suppose that the available simulation datasets are stored in the matrix $\boldsymbol{Y} = \{\boldsymbol{y_i}\}_{i=1}^{I} \in \mathbb{R}^{I\times a}$. The matrix $\boldsymbol{Y}$ can be decomposed through singular value decomposition as shown in equation (2.1):

$$\boldsymbol{Y} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{V}^T, \tag{2.1}$$

In this equation, $\boldsymbol{U} \in \mathbb{R}^{J\times r}$ and $\boldsymbol{V} \in \mathbb{R}^{a\times r}$ are orthogonal matrices, $\boldsymbol{\Lambda} \in \mathbb{R}^{r\times r}$ is a diagonal matrix that is composed of $r = \min(I, a)$ elements along its diagonal in decreasing order, and T is a transposition operator. Depending on the user defined accuracy for maintaining the percentage of information in matrix $\boldsymbol{Y}$, the first D singular values and their corresponding columns in $\boldsymbol{U}$ and $\boldsymbol{V}$ can be used to approximate $\boldsymbol{Y}$ ($D \ll r$). The POD bases $\boldsymbol{\phi}$ are given by the first D columns of $\boldsymbol{V}$, as shown in equation (2.2):

$$\boldsymbol{\phi} \coloneqq [\boldsymbol{\varphi}_1, \boldsymbol{\varphi}_2, \dots, \boldsymbol{\varphi}_d, \dots, \boldsymbol{\varphi}_D\ ] \in \mathbb{R}^{a\times D}. \tag{2.2}$$

Each of the POD bases $\boldsymbol{\varphi}_d = [\varphi_d(k_1), \varphi_d(k_2), \dots, \varphi_d(k_a)]^T$ represents a different dominant feature of $\boldsymbol{Y}$over the time interval $k \in [k_1, k_a]$. Each offline simulated results contained in $\boldsymbol{y}_i$ can be approximated by a linear combination of these POD bases as shown in equation (2.3):

$$\boldsymbol{y}_i \approx \sum_{d=1}^{D} \alpha_{id} \varphi_d(k), \quad k \in [k_1, k_a], \tag{2.3}$$

Here, α_{id} is the coefficient of the dth POD basis for the ith training data $\boldsymbol{y}_i$. The value of α_{id} can be obtained by projecting $\boldsymbol{y}_i$ onto the POD bases $\boldsymbol{\varphi}_d$, as shown in equation (2.4):

$$\alpha_{id} = \boldsymbol{y}_i \cdot \boldsymbol{\varphi}_d, \tag{2.4}$$

Thus, each offline simulated response $\boldsymbol{y}_i$ can be approximated with the obtained POD bases $\boldsymbol{\phi}$ and their corresponding low-dimensional coefficient row vector $\boldsymbol{\alpha_i} = [\alpha_{i1}, \alpha_{i2}, \dots, \alpha_{id}, \dots, \alpha_{iD}] \in \mathbb{R}^D$.

2.3.2 Gaussian Processes (GPs)

In order to characterize the variance of the dynamical responses over the different system inputs $\boldsymbol{x}$, a row vector of random variables, $\boldsymbol{z}(\boldsymbol{x}) = [z_1(\boldsymbol{x}), z_2(\boldsymbol{x}), \dots, z_d(\boldsymbol{x}), \dots, z_D(\boldsymbol{x})] \in \mathbb{R}^D$, is defined and used to statistically model the coefficients of the POD bases $\boldsymbol{\phi}$. Thus, for any given system input, the system response (or output) $y_k = f(\boldsymbol{x}, k)$ can be approximated though a linear combination of POD bases with stochastic coefficients $\boldsymbol{z}(\boldsymbol{x})$ as shown in equation (2.5):

$$f(\boldsymbol{x}, k) \cong \hat{f}(\boldsymbol{x}, k) = \sum_{d=1}^{D} z_d(\boldsymbol{x}) \varphi_d(k), \quad k \in [k_1, k_a]. \tag{2.5}$$

Here, $\hat{f}(\boldsymbol{x}, k)$ is used for the estimated system response. The coefficients of each POD basis $z_d(\boldsymbol{x})$ are independently modeled through GPs, which are trained through the obtained coefficients' datasets from the previous step $\{\boldsymbol{x}_i, \boldsymbol{\alpha}_i\}_{i=1}^{I}$. GPs [33-36] are a class of probabilistic models, which can be used to specify distributions over function spaces. It is assumed that the output responses over the whole design space follow a multivariate distribution such that the output response at a specific design configuration

follows a univariate Gaussian distribution. Here, each coefficient $z_d(\boldsymbol{x})$ of the dth POD basis is assumed to be a multivariate Gaussian distribution over multiple input configurations, and $z_d(\boldsymbol{x})$ at a given input is a univariate Gaussian distribution. The selections of the mean and covariance functions are based on the training data. A maximum likelihood estimation of the GP model parameters is carried out. Thus, in total, *D* number of GP models are trained. These GP models along with the POD bases obtained in previous steps are stored offline and later used online. As will be shown, the GP models and the POD bases can be used to efficiently and directly predict the system's dynamical response from equation (2.5). These dynamical response predictions along with their uncertainties will be treated as prior distributions for online updating.

2.3.3 Particle Filter (PF)

Sequential data assimilation can help improve model prediction quality by taking advantage of real-time observations, i.e., sensor measurements. For a linear stochastic-dynamical system, it is possible to derive an exact analytical expression for recursive calculations of posterior distributions by using the Kalman filter [37]. This method has been extended to deal with nonlinear systems by linearizing the state-space model, as with the Extended Kalman Filter (EKF) [38,39]. However, EKF suffers from a high computational expense in propagating the covariance error for large-scale problems, and is susceptible to instability [38, 40]. Instead, the Ensemble Kalman Filter (EnKF) has been proposed to approximate state distributions through a Monte Carlo approach [41-43]. Unlike the Kalman filter, with the PF [32], one relaxes the assumption that the

state variables are Gaussian distributed. Thus, one can handle the propagation of non-Gaussian distributions through nonlinear models.

To dynamically update the multi-step-ahead predicted system states by PF, one requires estimation of the full posterior distribution of the state variables, given the sensor measurement, $p(y_{k_1}, y_{k_2}, \dots, y_{k_a}|s_k)$, wherein p is the probability operator. The computational expense increases exponentially as the dimension of the prediction horizon grows, and thus may be inefficient for real-time usage. Here, PF is proposed to integrate with POD to reduce the computational complexity. One avoids inferring the probability density function (pdf) of the system states $\boldsymbol{y} \in \mathbb{R}^a$, but recursively updates the pdf of low-dimensional coefficients $\boldsymbol{y} \in \mathbb{R}^a$ $(D \ll a)$ given the sensor measurement s_k. A full explanation of the PF algorithm is as follows.

In PF, the estimated prior pdf of the coefficients is discretely represented by N number of particles, wherein the prior pdf is estimated through GP that is indicated as $\boldsymbol{z}_1^n$ for $n = 1, \dots, N$ and $\boldsymbol{z}_1 \in \mathbb{R}^D$. Here, the subscript of $\boldsymbol{z}_1$ is used to obtain the estimated coefficients at time $k = k_1$. Meanwhile, these particles are equally weighted initially as $w_1^n = 1/N$ and can be propagated into the state space via equation (2.6) as an a priori approximation of system dynamical response:

$$\hat{f}(\boldsymbol{x}, k) = \sum_{d=1}^{D} z_{1,d}^n(\boldsymbol{x})\varphi_d(k), \quad k \in [k_1, k_a]. \tag{2.6}$$

Then, during time step k when a sensor measurement s_k (for $k > k_1$) is available, a filtering posterior pdf of the coefficients can be approximated by using equation (2.7),

$$p(\boldsymbol{z}_k|s_k) \approx \sum_{n=1}^{N} w_k^n \delta(\boldsymbol{z}_k - \boldsymbol{z}_{k-1}^n), \tag{2.7}$$

where w_k^n is the posterior weight for the nth particle at time step k, and $\delta(\cdot)$ is the Dirac delta function. The first step in obtaining the posterior weights involves the calculation of the likelihood for each of the particles given the sensor measurement at time step k, as shown in equation (2.8) and equation (2.9):

$$s_k = g(y_k) + v_k, \tag{2.8}$$

$$p(s_k|\boldsymbol{z}_{k-1}^n) = \frac{L(s_k|\boldsymbol{z}_{k-1}^n)}{\sum_{n=1}^{N} L(s_k|\boldsymbol{z}_{k-1}^n)} = p(s_k - g(y_k^n)|R_k). \tag{2.9}$$

In equation (2.8), p is the probability operator, s_k is the measurement of the system response $g(y_k)$ with the assumed error v_k and the distribution of the sensor measurements is represented by R_k. In equation (2.9), the statistical properties R_k of the model prediction error $s_k - g(y_k^n)$ are assumed to calculate the probability of a sensor measurement through the likelihood function $L(s_k|\boldsymbol{z}_{k-1}^n)$. This likelihood function depends on the distribution of the assumed error v_k. By utilizing the normalized likelihood for each particle, the posterior weights can be obtained by equation (2.10):

$$w_k^n = \frac{w_{k-1}^n \cdot p\left(s_k - g(y_k^n)\middle|R_k\right)}{\sum_{n=1}^{N} w_{k-1}^n \cdot p\left(s_k - g(y_k^n)\middle|R_k\right)} \tag{2.10}$$

In this equation, w_{k-1}^n is the prior weight from the previous time step. At this point, a new weighted sample of the parameters can be used to estimate a discrete weighted approximation to the filtering posterior parameters, as well as state variables through equation (2.6).

As the weights of the discrete samples are updated, it is necessary to check the effectiveness of the samples since weight degradation may occur if the uncertainty of

the system is too large and too few samples have meaningful weights. The occurrence of the weight degradation can be detected from equation (2.11)

$$N_{eff} = \frac{1}{\sum_n^N (w_k^n)^2}, \tag{2.11}$$

where N_{eff} is a threshold of the minimum effective sample size that can be defined by a user to indicate the occurrence of degradation.

To address the degradation, a resampling procedure or systematic resampling [44] is selected to generate new samples $z_{k-1-resamp}^n$ in the parameter space. Systematic resampling is the most widely used algorithm in the literature due to how easy it is to implement and how it outperforms other resampling schemes in most cases [45]. Although basic resampling of states appears to be sufficient due to dynamic properties, parameter particles must be perturbed within a small neighborhood to avoid sample impoverishment [46],

$$\boldsymbol{z}_k^n = \boldsymbol{z}_{k-1-resamp}^n + \boldsymbol{\varepsilon}_k^n \quad \boldsymbol{\varepsilon}_k^n \sim N(0, b\sigma_{k-1}^z) \tag{2.12}$$

Here, $N(0, b\sigma_{k-1}^z)$ is a random sample from the Gaussian distribution with zero mean and variance $b\sigma_{k-1}^z$. The term $b\sigma_{k-1}^z$ is the product of the variance of the prior parameter from the last time step, σ_{k-1}^z, and a value, b, which is set in advance by a user.

Thus, the pdf of low dimensional parameters $\boldsymbol{z}_k$ are iteratively improved. Through the integration of these updated low dimensional parameters with extracted offline features, the model predictions can be dynamically enhanced. Meanwhile, the PF can be used to assimilate sensor measurements in an efficient manner, since the computational complexity of calculating multi-step-ahead system responses $\boldsymbol{y} \in \mathbb{R}^a$ is reduced to a subset of low dimensional parameters $\boldsymbol{z} \in \mathbb{R}^D (D \ll a)$.

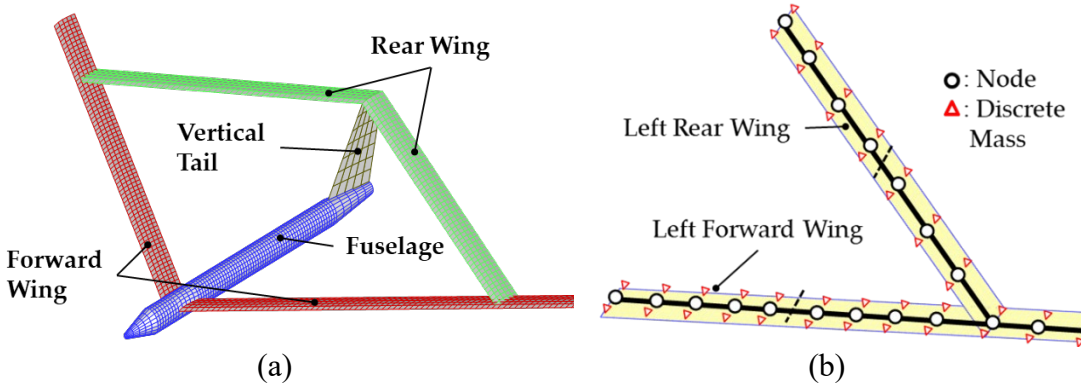
In the next section, an illustrative example, demonstrating the effectiveness of the proposed REC-MIMO strategy, will be carried out by using aeroelastic simulations of a joined-wing SensorCraft.

2.4 A Case Study: Predicting Nonlinear Aeroelastic Behavior of SensorCraft

A case study for predicting the nonlinear aeroelastic responses of a joined-wing SensorCraft [1, 2] is presented to demonstrate the proposed approach. Here, the detailed description of the case study is first presented in Section 2.4.1. This is followed by an application of the proposed framework step-by-step in Section 2.4.2. Finally, the results and discussions are provided in Section 2.4.3.

2.4.1 Description of the Case Study

The joined-wing SensorCraft is an experimental aircraft designed with thin and highly flexible wings to enhance mission efficiency. However, such a flexible structure interacting with fluid flow is susceptible to aeroelastic instabilities, which if not carefully addressed can cause structural failure of the aircraft. To address such effects, one requires effective long-term (multi-step ahead) prediction of the aircraft behavior during its operation in real-time to avoid possible failures. Thus, to realize a safe flight envelope for the SensorCraft, the proposed framework is applied to provide efficient long-term predictions to determine the system's aeroelastic stability for various flight conditions. More detailed descriptions of the joined-wing SensorCraft aeroelastic model can be found in the literature [47-49].



(a) (b)

Figure 2.3 Aeroelastic simulation: (a) aerodynamic grid representation of the joined-wing SensorCraft, and (b) beam representation of the wings of joined-wing SensorCraft.

In this case study, the offline datasets for estimating nonlinear aeroelastic behavior are computed by a high-fidelity aeroelastic simulation model, which is executed through a co-simulation scheme [50]. In this scheme, a coupled system is partitioned into subsystems that are simulated separately and then numerically integrated. Here, the co-simulation strategy involves the use of the Unsteady Vortex Lattice Method (UVLM) and the Finite Element (FE) method for capturing the aerodynamics and structural dynamics, respectively. The UVLM based aerodynamic model is used to simulate the aerodynamic loads acting on the aircraft. Then with these computed aerodynamic loads, the FE structural model is used to simulate the motions resulting from the aerodynamic forces acting on the aircraft's wings. In Figure 2.3(a), the aerodynamic mesh of the joined-wing aircraft is shown. In Figure 2.3(b), a beam model of one of the wings of the aircraft is shown. More details on the co-simulation strategy along with the UVLM can be found in references [49, 50].

The inputs to the aeroelastic simulator are representative of three different physical flight parameters. These parameters are the freestream velocity's magnitude (airspeed),

V, angle of attack, AoA, and atmospheric air density, ρ. The airspeed is affected both by the prescribed speed of the aircraft and also by head or tail winds. The angle of attack considered is the relative angle of the aircraft made with respect to the head wind direction. Variations in the altitude of operation can cause the air density to vary. The outputs of the simulator are the modal displacements (from which nodal displacements can be also obtained) corresponding to the first ten vibration modes and their associated derivatives with respect to time. For the case study, here, only the response (displacement) of the first mode has been chosen to demonstrate the implementation of the proposed approach. In order to study the aeroelastic stability, the simulation time was set sufficiently high (74,882 time steps for each scenario) for the initial transient behavior to die out.

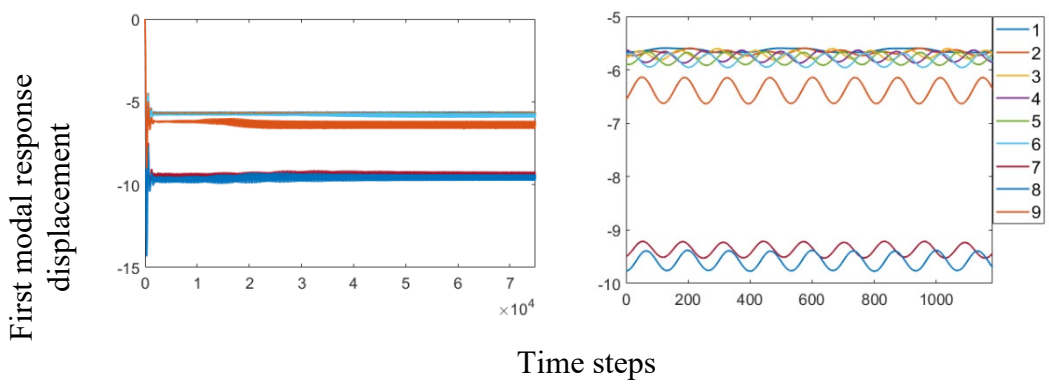
The available datasets are composed of nine simulated results at different operating conditions within the post-flutter regime (after the onset of flutter) as shown in Table 2.1. These nine cases have been intentionally set to have the same environmental conditions (e.g., the same air density). This is done so, as the goal of this case study is to interpolate the nonlinear aeroelastic response and possibly identify the flutter boundary (e.g., onset of flutter speed) for a given environment condition. However, the algorithm can be used to predict for any given air density. In Figure 2.4(a), the time dependent modal displacements for those nine conditions are depicted. The simulated responses have two stages. The initial stage is dominated by transient effects, which persist for approximately 10,000 time steps. The ensuing phase is in steady state oscillation. During this phase, the nature of the oscillation, in particular the amplitude, can be considered as a measure of the response stability. In this study, the author is

primarily concerned with predicting the presence of flutter instability. To this end, it is ensured that all transient effects have disappeared in the response studies; so, approximately, the final ten periods of oscillation are extracted, as illustrated in Figure 2.4(b). Amongst these nine truncated simulation results, each one of them will be used for the online performance to be estimated, while the other eight sets of data are treated as available offline datasets.

For the chosen case study, the sensor is assumed to measure the modal displacement of the first response mode. In practice, sensor point displacements can be associated with modal displacements making this a reasonable assumption. Here, sensor data are synthetically generated from a test data set by adding to simulated data a systematic Gaussian noise with zero mean and a standard deviation of 0.02 nondimensional displacement (6% of the average amplitude), but the prior knowledge of the sensor noise is not used in this case study. Instead, the author assumes a large level of sensor noise, which is two times the actual standard deviation (0.04 nondimensional displacement). Towards the end of the case study, the effects of sensor measurement bias and different levels of sensor noise on the prediction accuracy are investigated in Section 2.4.3.

Table 2.1 Nine operating conditions used in aeroelastic simulations.

Simulation Data Set	Airspeed V (m/s)	Angle of Attack AoA (°)	Air Density $\rho\,(kg/m^3)$
1	160	5	0.1152
2	161	5	0.1152
3	162	5	0.1152
4	163	5	0.1152
5	164	5	0.1152
6	165	5	0.1152
7	170	10	0.1152
8	175	10	0.1152
9	180	5	0.1152



(a) (b)

Figure 2.4 Modal displacements of the first mode responses obtained through simulations of 9 operating conditions: (a) over full simulation time and (b) after steady state is reached. All quantities are nondimensional.

2.4.2 Application of the Proposed Approach

In this section, the step-by-step prediction procedure for estimating the first modal response displacement is presented; the procedure is shown for one flight condition: $V = 164m/s$, $AoA = 5^o$, and $\rho = 0.1152kg/m^3$. The final prediction results for all the nine cases are presented in the following section.

The first step of the proposed approach is to apply POD to extract dominant features from the 8 sets of offline simulation results. As shown in Figure 2.5(a), in this graph, the number of POD bases versus the accumulated percentage of response from the

original data is depicted. Here, the number of POD bases is chosen to be five, which is sufficient to accurately capture 93.2% of the system's response. This is attained by calculating the sum squares of the first five singular values over all the singular value components. The corresponding five POD bases are depicted in Figure 2.5(b). Then, by using equation (2.4) to project the original simulation data onto the subspace composed of five POD bases, a set of coefficients $\{\boldsymbol{x}_i, \boldsymbol{\alpha}_i\}_{i=1}^{8}$ corresponding to different input flight conditions $\boldsymbol{x}$ are obtained.

In the second step, the GPs are used to characterize the variance of the coefficients over varying flight conditions, which are trained by the obtained coefficients data sets $\{\boldsymbol{x}_i, \boldsymbol{\alpha}_i\}_{i=1}^{8}$ from the previous step. Since each element of the coefficients row vector $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \dots, \alpha_5]$ is assumed to be independently modeled, 5 GP models are constructed. These GP models combined with the POD bases in the form of equation (2.8) are stored offline for online usage, which can effectively make long-term prediction as prior information for later data assimilation.

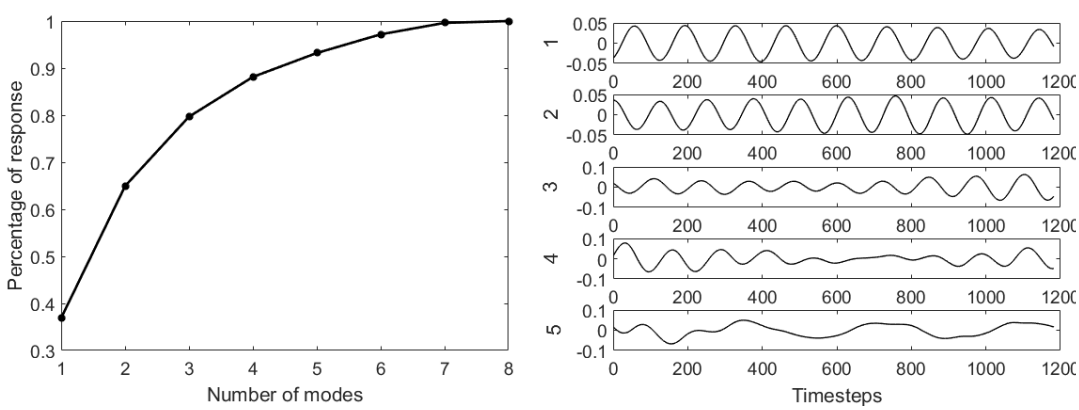


Figure 2.5 (a) Number of eigenmodes versus the percentage of response captured, and (b) associated temporal behavior.

During a real-time operation, with the collection of sensor measurements (green triangle in Figure 2.6), the prior prediction from the offline models can be improved by using the PF. An example of the assimilation process is shown in Figure 2.6, wherein the fit over time is shown by comparing the simulated response (red line in Figure 2.6) provided by the test dataset with the predicted response (blue line in Figure 2.6) obtained from the proposed approach. In Figure 2.6, it shows the state of the prediction at times k_1, k_8, k_{24}, and k_{100}. In Figure 2.6(a), it presents the initial long-term prediction from time k_1 up to k_{1183}, where there is relatively large variance in prediction (grey area) and the prediction accuracy decays when approximating for more than 500 time steps. The estimate can be seen to continuously improve with the assimilation of more sensor measurement data. After the first 8 time steps, there is decrease in prediction uncertainty and increase in prediction accuracy up to around k_{600}. By the 24th time step, the variance has dropped substantially, and after the 100th step, the difference between the simulated response and model forecasting is not discernible. Hence, for around 100-time steps, the proposed approach is able to capture the aeroelastic behavior with comparable accuracy to the high-fidelity simulation model.

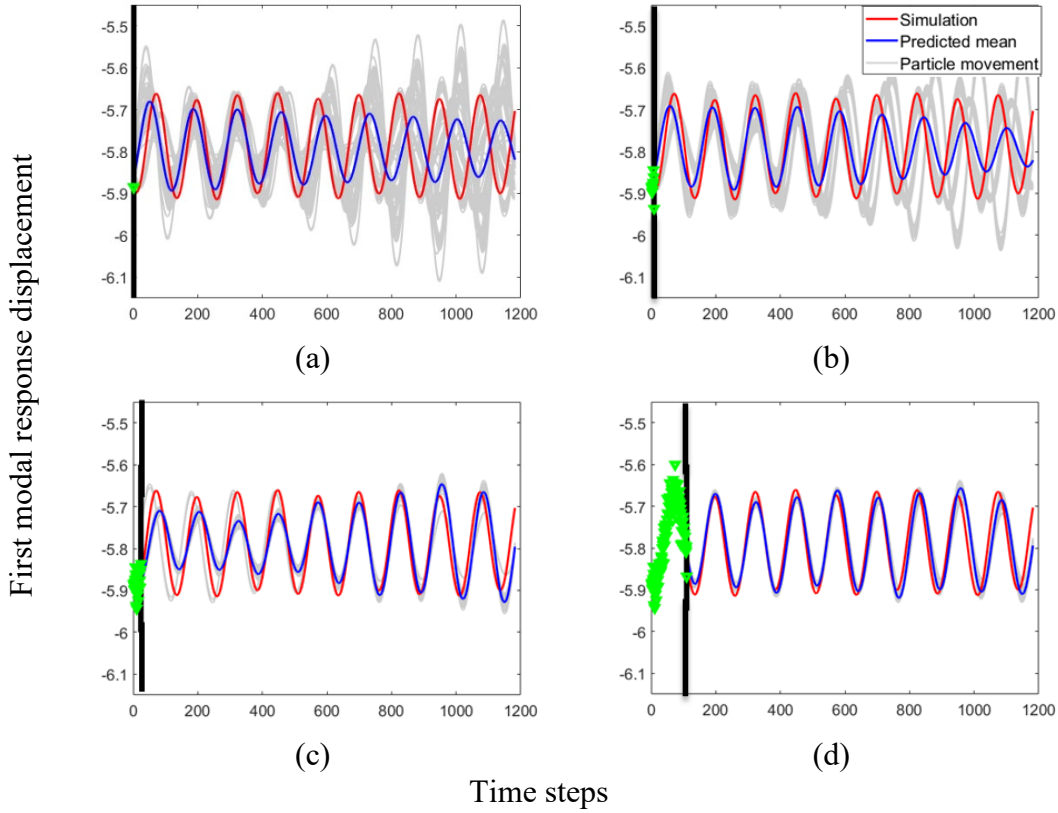


Figure 2.6 Online data assimilation process: model prediction versus simulation at different times (a) k_1, (b) k_8, (c) k_{24}, and (d) k_{100} . All quantities are nondimensional.

2.4.3 Results and Discussion

Following the cross-validation process, each of the nine cases will be used to imitate an online operation to be estimated. The final predictions obtained after data assimilation are shown in Figure 2.7, where the x-axis indicates the time steps of prediction horizon, and y-axis displays the first modal displacements. In these Figures, the estimated displacements (blue line in Figure 2.7) are compared against the simulated results (red line in Figure 2.7) to prediction accuracy. For the majority of the cases, the prediction matches discernibly well and it converges to a reasonable accuracy level with simulation at an early stage. However, in scenarios 1-3, one has less than

ideal fits. The major cause is assumed to be the exceptionally distinct modal displacement contained in these datasets, as they are closer to flutter boundary (onset of flutter speed $156\,m/s$). Thus, the obtained POD basis may be insufficient in representing the dominant feature in these cases. However, with the information collected from sensor measurements, the predictions can be dynamically enhanced to capture these distinct modal displacements through the proposed framework. Additionally, the number of simulated datasets in this case study is limited. With more training sets, the chances of having scenarios with similar oscillations modes would increase, which would in turn help further enhancing the model predictive abilities.

The root mean squared error (RMSE), calculated by using equation (2.13), for each of the nine scenarios is displayed in Table 2.2.

$$RMSE = \left(\frac{\sum_{k=k_1}^{k_a} (\hat{y}_k - y_k)^2}{a} \right)^{1/2} \tag{2.13}$$

Here, $\hat{y}_k$ is the predicted modal displacement, y_k is the simulated displacement at time step k, and a is the total number of time steps for each operating scenario.

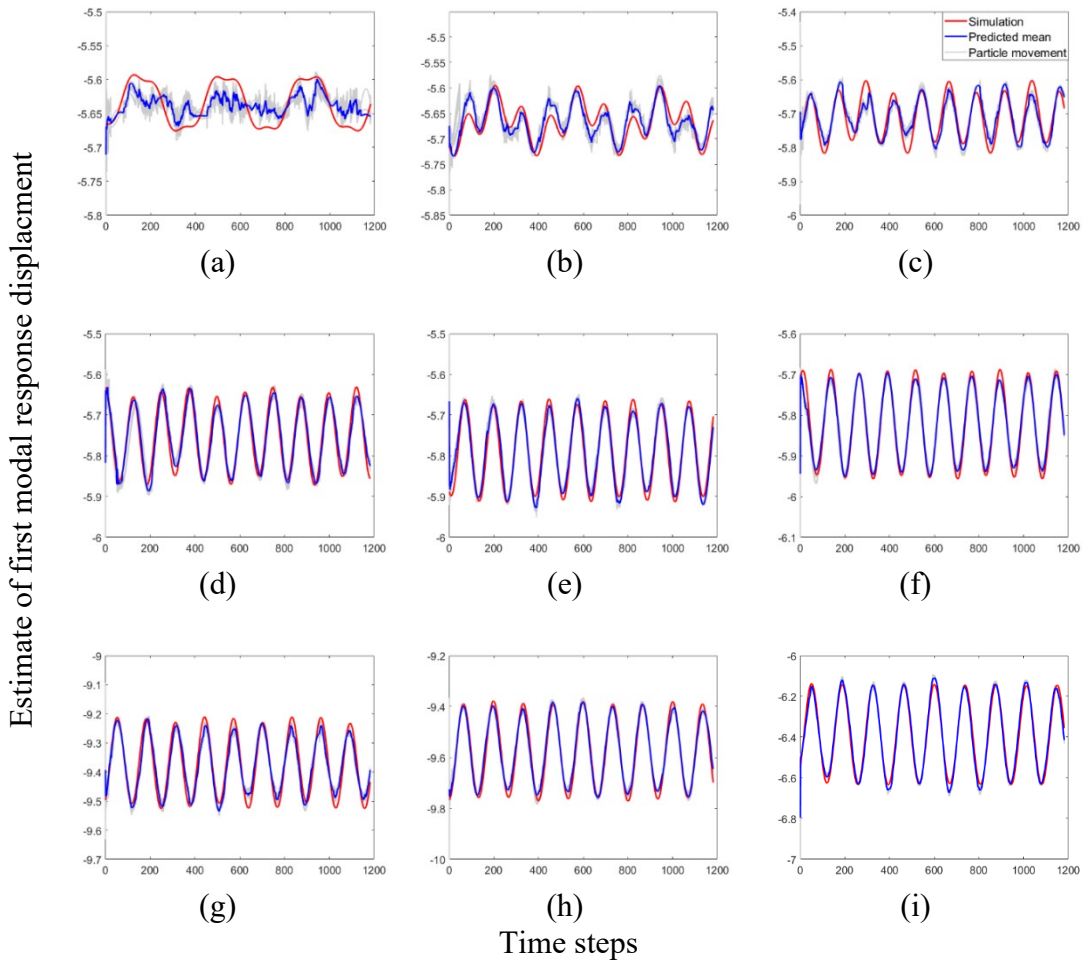


Figure 2.7 Final model estimations versus simulations for different test scenarios: (a - i) Scenarios 1 – 9.

Table 2.2 RMSE for 9 operating scenarios.

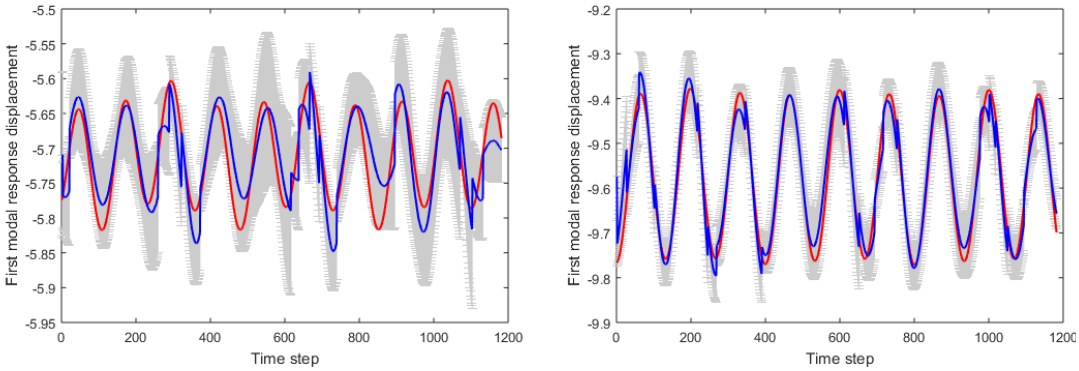
Inputs	1	2	3	4	5	6	7	8	9	Mean
RMSE	0.020	0.017	0.022	0.016	0.016	0.011	0.026	0.019	0.022	0.019

To further study the robustness of the proposed framework, the effects of sensor measurement bias and the noise level on the model prediction performance are investigated. Here, in addition to the white Gaussian noise contained in a sensor measurement, 10% measurement bias (0.26 nondimensional displacement) has been added to assess the sensitivity of the proposed framework to the measurement error.

Following the application of the proposed framework, the final predictions for each of nine cases can be obtained. Two representative prediction results have been selected to illustrate the influence of measurement bias on the prediction performance, as shown in Figure 2.8. One representative result is from test scenario 8 as illustrated in Figure 2.8(b). The predicted modal displacements still show noticeable agreement with simulation even in the presence of measurement error. However, in contrast to the results without measurement bias in Figure 2.7(h), the prediction uncertainties augment and persist for a longer time horizon. Another representative result is for the test scenario 3 as displayed in Figure 2.8(a). Comparing with estimates given in Figure 2.7(c), the inclusion of measurement error induces more prediction spikes, as well as increases the prediction uncertainty. It is believed the major reason for this is the increase in the uncertainty in the assimilation process due to the measurement bias. As mentioned before, the offline expensive simulation might be inadequate in addressing the exclusive displacement pattern of case 3. It requires the integration of simulation with sensor measurements for capturing such dynamics. The measurement bias extends the number of assimilation cycles, and uncertainties in the resampling procedure. Additionally, different proportions of measurement bias (fixed noise), from 6% to 20% of the average amplitude, are used and the framework is tested for the prediction performance as shown in Figure 2.9(a). As shown in this figure, the RMSE rises with introduction of measurement bias. However, the prediction error is pretty robust with a certain range of measurement bias until it is over 20% of average amplitude. The predicted RMSE for different levels of sensor noise, with standard deviations from 2% to 30% of the average amplitude, are shown in Figure 2.9(b). A trend can be seen in

this figure, the increase of RMSE is much lower than the increase of sensor noise. Thus, it can be said that one can use the proposed approach to obtain relatively robust prediction comparable to the expensive simulation, even in the presence of sensor measurement bias and noise.

In this case study, the computing platform that was used is a Dell computer with a Windows 10 operating system, Intel Xeon CPU E3-1245 v5 3.50GHz, and 16.0 GB RAM. During the offline stage, initial predictions from the stored offline models were made within 0.01 seconds. For the assimilation of the sensor measurement data, each iteration takes approximately 0.01 seconds. In this case study, it takes around 100 assimilation iterations (one second in total) to attain a comparable accuracy to that of the high-fidelity aeroelastic simulator, as shown in Figure 2.6(d). This is a major reduction in computational expense compared to the simulation model, which takes 30 hours to simulate the aeroelastic behavior for each operating condition. Additionally, it is envisioned that the proposed approach may also be used in conjunction with motion estimation schemes such as those presented for systems with flexible wings (e.g., [51]).



(a) (b)

Figure 2.8 Representative model prediction with 10% sensor measurement error for test scenarios: (a) scenario 3, and (b) scenario 8.

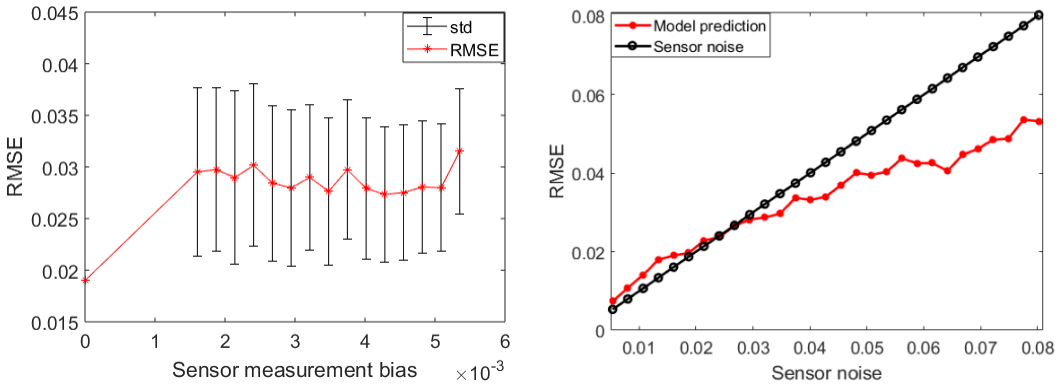


Figure 2.9 Mean RMSE of model predictions at different level of (a) measurement bias, and (b) sensor noise.

2.5 Summary

In this chapter, a new dynamical multi-step-ahead prediction strategy (REC-MIMO) has been developed and presented. This strategy is designed to predict single-response system's behaviors in real-time through an efficient integration of simulation data (offline) and sensor measurement data (online). During the offline stage, a MIMO model is constructed to directly forecast the system's dynamical response for varying system inputs and time steps, which is achieved through a combination of POD and GPs. Then, during the online phase, PF is utilized to recursively update the prior predictions with the collected sensor measurement data. With the results obtained from the aeroelastic simulations of the experimental joined-wing SensorCraft, the effectiveness of the proposed framework for prediction of aeroelastic responses has been demonstrated. These multi-step-ahead predictions can be used to identify the precursor of the aeroelastic instabilities with comparative accuracy to a high-fidelity simulation model, while significantly reducing the computational expense. In addition, the further investigation is conducted to study the effects of measurement bias and noise on the model performance. Based on the case study, it has been revealed that the

prediction accuracy is relatively robust even in the presence of measurement bias and noise.

In the next chapter, the second research question will be presented, which is an extension of the prediction system behavior strategy presented in this chapter with applications in evaluating various system's designs and operations.

Chapter 3: Data-Driven Prediction for Design and Operation: Application to Aeroelastic Response and Robotic Appendage

In this chapter, the outcome of the investigation due to Research Question 2 is presented, i.e., *how can we integrate data obtained offline from high-fidelity simulations with limited, online multi-sensor measurements (or physical experiments) to (i) predict multiple output responses (representing behavior) of a system, multiple time steps ahead, and (ii) use the prediction to evaluate various system designs and/or operations?*

The material for this chapter is organized based on the author's previous articles[3,4]. Accordingly, the description starts with a nomenclature section specifically tailored for this chapter. Next, the formulation of a prediction approach is presented. The prediction approach in this chapter is an extension of that presented in Chapter 2. For example, while in Chapter 2 the prediction was constructed from the data obtained using a high-fidelity, single-response simulation, in this chapter, the prediction is obtained from a high-fidelity multi-response simulation. Moreover, while the (synthetic) data in Chapter 2 was generated from a single sensor, in this chapter, the synthetic data is generated for a sparse number of sensors (or from a limited number of physical experiments).

---

[3] Zhao, X., Azarm, S. and Balachandran, B. (2021) "Online Data-Driven Prediction of Spatio-Temporal System Behavior Using High-Fidelity Simulations and Sparse Sensor Measurements." *Journal of Mechanical Design*, 143(2): 021701.

[4] Zhao, X., Azarm, S. and Balachandran, B. (2018) "Dynamic Data-Driven Spatiotemporal System Behavior Prediction with Simulations and Sensor Measurement Data." *ASME 2018 International Design Engineering Technical Conferences and Design Automation Conference,* vol(2B): V02BT03A060, Quebec City, Canada, August 26-29.

In Section 3.1, a description of the problem statement together with corresponding assumptions are provided. Then, in Section 3.2, the background literature is reviewed and gaps are identified. Next, in Section 3.3, an extension to the dynamic data-driven prediction approach from Chapter 2 is presented. The approach is constructed by using and fusing data from different, including those obtained from the high-fidelity multi-response system simulation, and from a sparse number of sensors (or spare number of physical experiments). Following this, the proposed prediction approach is used for evaluating a variety of system designs and/or operational conditions. Next, in Section 3.4, the applicability of the proposed prediction approach is demonstrated step-by-step using a numerical example, and further evaluated by two engineering examples. In the first engineering example, an application of the approach using a multi-response aeroelastic prediction for a SenorCraft is presented, wherein by changing sensor locations, the predictive capability of the proposed approach is investigated. The second example is on using the prediction approach for evaluating a variety of designs and operational conditions for robotic appendage motions, based on a combination of data collected from high-fidelity simulations and limited physical experiments obtained from the literature. Finally, a summary for this chapter is provided in Section 3.5.

Nomenclature for Chapter 3:

a	total number of time steps
$a_{j,k}$	a component of parameter matrix $\boldsymbol{A}$ with respect to each index
$\boldsymbol{A}$	random variables of parameter matrix
$\boldsymbol{d_j}$	coordinates of the j-th spatial location
$\boldsymbol{f}(\cdot)$	system's dynamic function
i	index for simulated dataset
I	total number of simulated datasets (or training data)
j	index for spatial location
J	dimension of multi-response system behavior
k	index for time step
l	dimension of measurement data (sensors or physical experiments)
$L(\cdot)$	likelihood function
m	dimension of system inputs
$\boldsymbol{M}$	sparse diagonal sensors measurement matrix
N_p	total number of particles
p	index for particle
$P(\cdot)$	probability function
r_1	rank for spatial dimension

r_2	rank for temporal dimension
$R_{j,k}$	variance of sensor measurement at *j*-th location and time step *k*
R_k	variance vector of sensor measurements at time step *k*
$s_{j,k}$	sensor measurement at *j*-th location and time step *k*
$\boldsymbol{s}_k$	sensor measurements at time step *k*
v	airspeed
w	importance weight for a particle (or sample point)
$\boldsymbol{u}_i, \boldsymbol{v}_j, \boldsymbol{w}_k$	*i*-th, *j*-th and *k*-th eigenvectors for eigen matrices $\boldsymbol{U}$, $\boldsymbol{V}$ and $\boldsymbol{W}$ respectively
$\boldsymbol{U}, \boldsymbol{V}, \boldsymbol{W}$	eigen matrices for tensor χ
$\widetilde{\boldsymbol{V}}$, $\widetilde{\boldsymbol{W}}$	truncated eigen spatial and eigen temporal matrices
$\boldsymbol{x}$	a vector of variables for system design
$y_{j,k}$	system behavior at *j*-th location and time step *k*
$y_{j,k}^p$	*p*-th particle value for system behavior at *j*-th location and time step *k*
$\boldsymbol{y}_k$	system behavior for all spatial locations at time step *k*
$\boldsymbol{Y}$	system behavior at all spatial locations and time steps
$\boldsymbol{Y}_i$	spatio-temporal system behavior for *i*-th system input $\mathbf{x}_i$
$\boldsymbol{X}$	vector of system input
$\mathcal{A}$	parameter tensor

$\delta(\cdot)$	dirac delta function
ξ	decomposed core tensor
ρ	air density
ω	stride frequency
χ	offline simulated (or training) tensor data

ACRONYMS

AoA	angle of attack
DDDAS	dynamic data-driven application systems
EnKF	ensemble Kalman filter
GP	Gaussian process
HOSVD	higher-order singular value decomposition
NN	neural network
PCA	principal component analysis
pdf	probability density function
PF	particle filter
ROPF	reduced-order particle filter
SVD	singular value decomposition

3.1 Prediction for System Design and Operation: Problem Formulation and Assumptions

The goal in this problem is to provide efficient predictions for multi-response system behavior under a variety of system designs and/or operational conditions. A block diagram of the prediction problem for this chapter is depicted in Figure 3.1. Here, multi-response system behavior is represented by a vector $\boldsymbol{y}_k \in \mathbb{R}^J$ with the dimension of J, which is given by a vector of functions, $\boldsymbol{y_k} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{d}_j, k)$. Here, $\boldsymbol{x} \in \mathbb{R}^m$ is a vector of the system's inputs, which can representing system's design variables and/or system's operational condition; $\boldsymbol{d}_j, j = 1, \dots, J$, is a three-dimensional vector that consists of the coordinates of the system's j-th location; and k is an index for the time steps in the range $k \in [k_1, k_a]$, where k_1 and subscript a indicate the initial time step and the number of time steps of the prediction, respectively. The system's behavior function $\boldsymbol{f}(\cdot)$ is assumed to vary nonlinearly with respect to the system's input, space (or location) and time.

To estimate the multi-response system behavior, it is assumed that high-fidelity simulations can provide a set of simulated input-output data, $\{\boldsymbol{x}_i, \boldsymbol{Y}_i\}_i^I$, during an offline stage. Here, I is the total number of offline datasets, i is the index of each dataset and $\boldsymbol{Y} \in \mathbb{R}^{J \times a}$ is a matrix of the system's response for all system's spatial locations and time steps. It is assumed that during the online stage, sensors can provide measurements, $\boldsymbol{s}_k \in \mathbb{R}^l$, of the system's response for each time step, where l indicates the total number of sensors. The sensors are assumed to be sparsely located on the system and measure the system behavior y_k for each time step k. The locations of the sensors are indicated by $\boldsymbol{M} \in \mathbb{R}^{J \times J}$, a sparse diagonal matrix with each of its diagonal

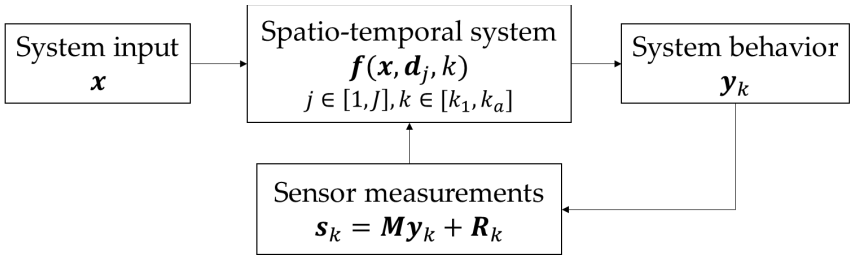


Figure 3.1 Block diagram of the problem.

element's binary values representing whether a sensor is located (with the value of one) at the system's j-th location or not (with the value of zero). For the sensors, the following assumptions are made: (i) there is no interference among the sensors, (ii) the number of sensors is assumed to be much smaller than the number of the system's spatial locations of interest ($l \ll J$), and (iii) all sensors are operational at all times, and are subjected to Gaussian or non-Gaussian white noises, where their noise variance vector (for all sensors) indicated as $\boldsymbol{R}_k$ for each time step k.

The sensor data or physical experiment data and available offline simulation datasets are combined using the proposed approach, as detailed in the next section, to predict the behavior of the system for any configuration of the system's design and operational conditions.

3.2 Related Literature Review

Several challenges exist in the online prediction of multi-response system behavior with data-driven approaches, especially for the system involves spatio-temporal behavior and for a variety of system designs and/or operational conditions. One main challenge is the management of uncertainty associated with sparsely distributed sensor measurements as well as the uncertainty introduced through inference from sensor data [52-56]. A second challenge is the existence of a complex spatio-temporal correlation

among responses [52-55]. This includes the spatial information, as well as the complex dependencies among time-varying multiple responses of the system. An additional challenge is when there is a limit on the available computational resources available and a possible need for online decision making [56].

Due to these challenges, traditional linear data-driven methods, for example, autoregressive moving average [57], often result in poor prediction of the system's behavior. Recently, several nonlinear data-driven prediction methods have been proposed, especially, based on (i) Neural Network (NN) [58-61], (ii) Bayesian estimation methods [34, 41-42, 62, 64-67], and (iii) functional decomposition methods [30-31, 68-70]. However, these recent methods may not be readily applicable either, as discussed next.

Firstly, deep learning based NN methods, e.g., Long Short-Term Memory [58] (LSTM), have been widely applied for time series predictions. The NN based methods are known to perform well in predicting complex dynamics for many applications, including road network traffic [58, 59], wind power [60], and so on. However, NNs usually require a large training set and have not been very reliable in handling the effect of data uncertainty [62, 63], e.g., due to sensor measurements. Moreover, NNs do not directly include spatio-temporal correlation for modeling the system behavior [62, 63].

Secondly, methods such as recursive Bayesian estimation can be used to dynamically improve model prediction with the incorporation of online measurements. For instance, in contrast with the Ensemble Kalman Filter (EnKF) method [41], by using the Particle Filter (PF) [32, 64], one can propagate non-Gaussian distributions through nonlinear models, but at the cost of increasing the computational expense. To

efficiently update a multi-step-ahead system prediction, an approach is proposed in the author's previous work [71], which integrates a Particle Filter (PF) method with proper orthogonal decomposition to reduce the dimension and computationally complexity of the data assimilation procedure, and thus made it possible for real-time usage. In addition, Gaussian Processes (GPs) [34, 65-67] have also been widely studied as Bayesian nonparametric methods for predicting system's dynamics under uncertainty. Although these Bayesian based methods are inherently able to handle measurement uncertainty, they suffer from exponential computational complexity as the dimension of the problem grows.

Thirdly, functional decomposition methods have received increased attention as they have been used to make complicated model prediction computationally tractable. Typically, methods in this category are hybrid, wherein an adaptive basis is formed from the data and then predictive methods are employed to forecast the evolution of individual components of the basis. For example, the data can be decomposed as sinusoidal wave bases by Fourier analysis [68], Haar wave bases through Wavelet transform [69], or eigenvectors in Principal Component Analysis (PCA) [30-31, 70]. Following that, a neural network can be employed to estimate the individual components. However, conventional linear decomposition methods, such as PCA, often become inadequate to address spatio-temporal data. In those methods, only one factor at a time is allowed to vary; for example, either time or space, and moreover the spatio-temporal correlation is ignored. On the other hand, the Higher-Order Singular Value Decomposition (HOSVD) method is considered as a generalization of the matrix singular value decomposition for multi-way arrays [72-74]. HOSVD has been

commonly applied to multi-model tensor data analysis, e.g., for face recognition [73], and real-time disease surveillance with complex spatio-temporal data streams [74].

In this chapter, the objective is to develop a fast, online, data-driven approach to predict spatio-temporal, multi-response system behavior for dynamical systems through the integration of high-fidelity simulation data (offline) and sensors measurement data (or physical experiment data). This chapter is an extension of the author's earlier work [71], as presented in Chapter 2.

In this chapter, the author addresses two new challenges, namely: 1) sparse sensors measurement data (or physical experiments), which include sensor measurement uncertainty as well as the uncertainty due to the inference from sensor data, and 2) multi-response simulation and spatio-temporal correlation among responses.

Based on the gaps identified, several contributions are made in this chapter. Firstly, the proposed approach can be used to reduce the dimension of the simulation data while still retaining the dependencies among time-varying responses obtained from the system's different locations. Secondly, in the proposed approach, the GP integrated with HOSVD is constructed as a surrogate model to estimate the nonlinear system's behavior. The surrogate model is stored offline and used to make an efficient prior prediction for use during the online stage. Thirdly, a data assimilation method, the Reduced-Order Particle Filter (ROPF), is proposed to dynamically improve the surrogate model prediction with sparse, noisy sensors measurement data, or sparse data collected from physical experiments. This technique can be used to propagate the model predictions as well as their uncertainties for a nonlinear model. Meanwhile, ROPF significantly reduces the computational cost compared to the traditional PF.

3.3 Proposed Framework

As shown in Figure 3.2, the proposed approach is composed of three main steps, Step 1 to Step 3, and can be divided into an offline and an online phase. The offline phase is used to construct an efficient surrogate model using training data generated by the high-fidelity simulation. In Step 1 (see Figure 3.2), the HOSVD method is used to obtain eigen spatial and eigen temporal matrices from the simulation data, and to represent the simulation data by a set of low-dimensional projected parameters. In Step 2, the GPs trained by the parameter dataset obtained from Step 1 are employed to model the variation of the projected parameters over different values of system inputs. These GP models along with the eigen spatial and eigen temporal matrices are stored offline for making prior spatio-temporal prediction during online operation. In Step 3, ROPF is utilized to assimilate sparse sensors measurement data to iteratively update the prediction of the projected parameters. By integrating the updated parameters with eigen spatial and eigen temporal matrices, an efficient update for spatio-temporal predictions can be obtained.

A detailed explanation for each step is presented in Sections 3.3.1-3.3.3. To clarify the notation, the symbols used for scalars, vectors, matrices, and tensors, as well as their products are defined in Table 3.1.

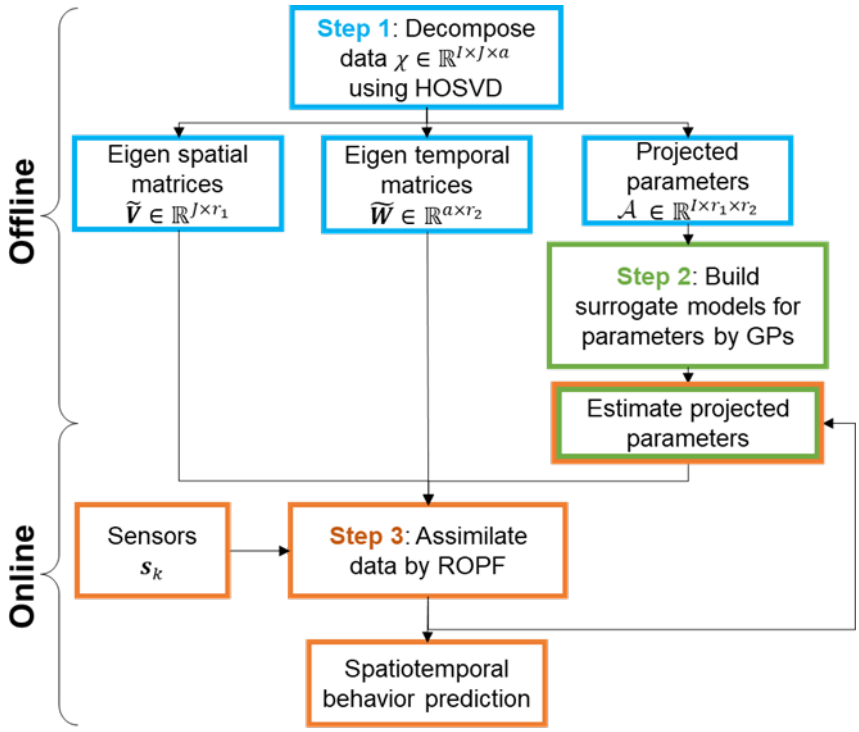


Figure 3.2 Framework for the proposed approach

Table 3.1 Definition of symbols and products

Symbol	Definition
$\chi, \boldsymbol{X}, \boldsymbol{x}, x$	Tensor, matrix, vector, scalar
$\chi \times_i \boldsymbol{X}$	Multiplication of tensor and matrix, which forms the i-th index of χ with second index of $\boldsymbol{X}$
$\chi = \boldsymbol{x}_i \otimes \dots \boldsymbol{x}_n \dots \otimes \boldsymbol{x}_n$	Tensor or outer product of vectors $\boldsymbol{x}_n \in \mathbb{R}^{I_n} (n = 1, \dots, N)$ yields a rank 1 tensor $\chi \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$

3.3.1 Higher-Order Singular Value Decomposition (HOSVD)

The approach presented in this section follows the ideas in a multilinear projection algorithm [72, 73], which is used in dimension reduction of images resulting from multiple factors. Here, HOSVD is used to reduce the dimension of the system output. HOSVD, which is a multilinear extension of the matrix SVD to higher-order tensors, is employed to analyze system behavior ensembles in which multiple variables, such as time, space, and system inputs, are allowed to vary. Let the tensor $\chi = \{\boldsymbol{Y}_i\}_i^I \in$

$\mathbb{R}^{I\times J\times a}$ represent an ensemble of offline simulated system behaviors. Here, a three-mode SVD is used to decompose the tensor χ, as shown in equation (3.1):

$$\chi = \xi \times_1 \boldsymbol{U} \times_2 \boldsymbol{V} \times_3 \boldsymbol{W} \tag{3.1}$$

where $\times_1 \boldsymbol{U}$, $\times_2 \boldsymbol{V}$, and $\times_3 \boldsymbol{W}$ denote multiplication of the core tensor $\xi \in \mathbb{R}^{I\times J\times a}$ and the eigen matrices $\boldsymbol{U} \in \mathbb{R}^{I\times I}$, $\boldsymbol{V} \in \mathbb{R}^{J\times J}$, and $\boldsymbol{W} \in \mathbb{R}^{a\times a}$, form the first, second and third indices of ξ with the second indices of $\boldsymbol{U}$, $\boldsymbol{V}$ and $\boldsymbol{W}$, respectively. Here, the core tensor ξ governs the interaction between the features of system inputs, space, and time represented in the eigen matrices.

The multilinear HOSVD of equation (3.1) can also be reformulated as the outer product, denoted by $\otimes$, of the eigenvectors as shown in equation (3.2):

$$\chi = \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=k_1}^{k_a} \xi_{i,j,k}\boldsymbol{u}_j \otimes \boldsymbol{v}_j^T \otimes \boldsymbol{w}_k^T \tag{3.2}$$

Where $\boldsymbol{u}_i$, $\boldsymbol{v}_j$, and $\boldsymbol{w}_k$, are the *i*-th, *j*-th, and *k*-th eigenvectors of the eigen matrices $\boldsymbol{U}$, $\boldsymbol{V}$ and $\boldsymbol{W}$, respectively. The superscript *T* is for transposition. Similarly to PCA, the tensor χ can be approximated by discarding the multi-linear eigenvectors and slices of the core tensors. The rank of the tensor can be determined by computing the multi-linear singular values, chosen based on the percentage of information needed or desired to be kept. Here, the ranks for the multi-dimensional tensor data can be different along different dimension. Let r_1 and r_2 be the respective ranks of tensor χ selected for the spatial and temporal dimensions. Thus, the truncated eigen spatial matrices are $\widetilde{\boldsymbol{V}} \in \mathbb{R}^{J\times r_1}$ and eigen temporal matrices $\widetilde{\boldsymbol{W}} \in \mathbb{R}^{a\times r_2}$. By a multi-linear projection of the tensor data χ onto the truncated eigen matrices, a set of low-dimensional projected parameter tensor $\mathcal{A} \in \mathbb{R}^{I\times r_1\times r_2}$ can be obtained:

$$\mathcal{A} = \chi \times_2 \widetilde{\boldsymbol{V}} \times_2 \widetilde{\boldsymbol{W}} \tag{3.3}$$

Here, the parameter tensor is composed of I number of parameter matrices. Each parameter matrix contains a set of coefficients for eigen spatial and eigen temporal matrices, which represents the interaction between spatial and temporal variability for the spatio-temporal system output corresponding to each system input. Note that, comparing against the conventional PCA, the multi-linear analysis has several advantages. First, it enables the representation of system behavior, regardless of different system inputs, by shared eigen spatial and eigen temporal matrices. Thus, HOSVD further reduces the dimension of the offline simulation data χ, while maintaining the spatio-temporal correlation. In this way, the simulated tensor data χ is approximated by a low dimensional parameter tensor $\mathcal{A}$, as in equation (3.4):

$$\chi \approx \sum_{j=1}^{r_1} \sum_{k=k_1}^{r_2} \mathcal{A}_{i,j,k} \boldsymbol{v}_j^T \otimes \boldsymbol{w}_k^T, \forall i \tag{3.4}$$

Here, $\mathcal{A}_{i,j,k}$ is a component of parameter tensor $\mathcal{A}$ with respect to each of its indices.

3.3.2 Gaussian Processes (GPs)

To model the variability of the spatio-temporal responses with respect to the system inputs, a matrix of random variables, $\boldsymbol{A}(\boldsymbol{x}) \in \mathbb{R}^{r_1 \times r_2}$, is defined to statistically model the projected parameters. Through an integration of these random variables with eigen spatial and eigen temporal matrices, the system's spatio-temporal output can be approximated, as in equation (3.5):

$$\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{d}_j, k) \approx \hat{\boldsymbol{f}}(\boldsymbol{x}, \boldsymbol{d}_j, k) = \sum_{j=1}^{r_1} \sum_{k=k_1}^{r_2} a_{j,k}(\boldsymbol{x}) \boldsymbol{v}_j^T \otimes \boldsymbol{w}_k^T \tag{3.5}$$

Here, $a_{j,k}$ is a component of the parameter matrix $\boldsymbol{A}$ with respect to each of its indices. Each of these parameter components is independently modeled through the GPs, which are trained by the estimated parameter tensor $\left\{\boldsymbol{x}_i, \mathcal{A}_{i,j,k}\right\}_i^I$ obtained from Step 1.

For the training of the GPs, the main computational burden is associated with the calculation and inversion of the covariance matrix. However, here the computational effort for the training has been mainly assigned to the offline stage. The constructed GP models along with the eigen spatial and eigen temporal matrices obtained in Step 1 are retained offline for online usage. Thus, the computational cost can be notably reduced for the online (real-time) phase. As a result, during the online stage, the spatio-temporal predictions can be obtained for any given system inputs. These predictions are treated as prior estimations and updated with sensors measurement data collected in Step 3.

3.3.3 Reduced-Order Particle Filter (ROPF)

This step consists of a data assimilation process, wherein the system's behavior estimation is recursively updated each time an observation becomes available. For the conventional PF, in order to dynamically update the spatio-temporal system response estimate, the estimate of the full posterior distribution of the system response variables $\boldsymbol{Y} \in \mathbb{R}^{J \times a}$, is necessitated as shown in equation (3.6):

$$P(\boldsymbol{Y}|\boldsymbol{s}_k) = \left(\begin{array}{cccc|c} y_{1,k_1} & y_{1,k_2} & \cdots & y_{1,k_a} & \\ y_{2,k_1} & y_{2,k_2} & \cdots & y_{2,k_a} & \\ \vdots & \vdots & \ddots & \vdots & \boldsymbol{s}_k \\ y_{J,k_1} & y_{J,k_2} & \cdots & y_{J,k_a} & \end{array}\right) \tag{3.6}$$

The computational expense for estimating the full posterior distribution can increase exponentially with an increase in both space and time dimensions. On the other hand, only a small number of sensor measurements $\boldsymbol{s}_k \in \mathbb{R}^l (l \ll J)$ is available, and the relationship between sensors measurement data $\boldsymbol{s}_k$ and system's full behavior $\boldsymbol{Y}$ is unknown. Thus, a direct implementation of the PF is also infeasible.

In this chapter, the ROPF method, proposed in the author's previous work [71], is extended to update the system's spatio-temporal estimates with a limited number of sensors measurement data. To do this, using ROPF, one takes advantage of the eigen spatial and eigen temporal matrices, as well as the model constructed in equation (3.5). Instead of inferring the distribution of the entire system behavior $\boldsymbol{Y} \in \mathbb{R}^{J \times a}$, the author estimates the posterior distribution of the projected parameter elements $P(\boldsymbol{A}|\boldsymbol{s}_k)$, $\boldsymbol{A} \in \mathbb{R}^{r_1 \times r_2}$ given the sensors measurement data, and thus this can significantly reduce the dimension and computational cost of the assimilation process.

A set of particles (or samples) are used to represent the prior/posterior distribution of the projected parameters. Each particle has a likelihood (or importance) weight assigned to it that represents the probability of that particle being sampled from the probability density function (pdf). Here, the prior pdfs of the projected parameters are represented by N_p number of particles $\boldsymbol{A}_1^p, p = 1, \dots, N_p$, wherein the prior pdfs are provided by the GPs in Step 2. Here, $\boldsymbol{A}_1$ is used to indicate the estimated parameters at time step $k = k_1$. Meanwhile, these particles are equally weighted as $w_1^p = 1/N_p$. The number of particles N_p is to be prespecified by the user. This number can be determined based on a trade-off between the estimated accuracy desired versus available budget on

the computational cost, as the larger number of particles can produce better precision at the expense of increasing the computational cost.

With respect to the collection of sensors measurement data $\boldsymbol{s}_k$ at each time step k, a filtering posterior pdf of these parameters $P(\boldsymbol{A}|\boldsymbol{s}_k)$ can be approximated, as in equation (3.7):

$$P(\boldsymbol{A}|\boldsymbol{s}_k) \approx \sum_{p=1}^{N_p} w_k^p \delta(\boldsymbol{A}_k - \boldsymbol{A}_{k-1}^p) \tag{3.7}$$

where w_k^p is the posterior weight for the p-th particle at time step k and $\delta(\cdot)$ is the Dirac delta function. To obtain the posterior weights, first, the author calculates the likelihood for each of the particles given the sensors measurement data at time step k, as in equation (3.8):

$$P\left(\boldsymbol{s}_k \middle| \boldsymbol{A}_{k-1}^p\right) = \frac{L\left(\boldsymbol{s}_k \middle| \boldsymbol{A}_{k-1}^p\right)}{\sum_{p=1}^{N_p} L\left(\boldsymbol{s}_k \middle| \boldsymbol{A}_{k-1}^p\right)} = \prod_{j=1}^{l} P\left(s_{j,k} - y_{j,k}^p \middle| R_{j,k}\right) \tag{3.8}$$

In equation (3.9), $s_{j,k}$ is the sensor measurement at j-th spatial location at time step k, and $R_{j,k}$ is its corresponding measurement variance. $y_{j,k}^p$ is the p-th particle value as a prediction of system behavior at j-th location and time step k, which is calculated through $\boldsymbol{A}_{k-1}^{\boldsymbol{p}}$ using equation (3.5). The statistical properties $R_{j,k}$ of the model prediction error $s_{j,k} - y_{j,k}^p$ are assumed to obtain the probability of a sensor measurement through the likelihood function $P(s_{j,k}|\boldsymbol{A}_{k-1}^{\boldsymbol{p}})$. Here, the sensor measurements are assumed to be independent. The likelihood function $P(s_{j,k}|\boldsymbol{A}_{k-1}^{\boldsymbol{p}})$ can be calculated through multiplication of the probability given each of the sensor measurements. Then, the posterior weights can be obtained through the normalized likelihood for each particle, as in equation (3.9):

$$w_k^p = \frac{w_{k-1}^p \times \prod_{j=1}^{l} P\left(s_{j,k} - y_{j,k}^p \middle| R_{j,k}\right)}{\sum_{p=1}^{N_p} w_{k-1}^p \times \prod_{j=1}^{l} P\left(s_{j,k} - y_{j,k}^p \middle| R_{j,k}\right)} \tag{3.9}$$

In equation (3.9), w_{k-1}^p is the prior weight from the previous time step. In this way, an updated weighted sample of the parameters can be used as a discrete approximation of the filtering posterior parameters.

One issue with the filtering process is that degradation in the estimate of weights may occur if the uncertainty among the particles is too large, and too few particles have meaningful weights. To address this issue, a systematic resampling process is used to avoid the sample degradation. The resampling algorithm is widely used in the literature due to easy implementation and outperforms other resampling schemes in most cases [44, 45]. Although the basic resampling of the system responses appears to be sufficient, the particles parameters are perturbed within a small neighborhood to avoid sample impoverishment [46].

3.4 Examples

In this section, the proposed approach is demonstrated by three examples. These include a numerical example and two engineering examples. Using the numerical example, in subsection 3.4.1, a step-by-step application of the proposed approach is demonstrated. Next, in subsection 3.4.2, the engineering example for prediction of a multi-response aeroelastic behavior for the joined wing aircraft is presented. Then, in subsection 3.4.3, an engineering example for prediction and evaluation of robotic appendage interaction with a terrain with granular material under different design and operational conditions is investigated.

3.4.1 Numerical Case Study

In this example, a system is defined by a simple function as its simulation model: $f(x,d,k) = xd^2 + x^2(d+1)k$. For this example, a prediction is to be made at two discrete values of: $d_1 = 0, d_2 = 1$, and for two-time steps $k_1 = 1, k_2 = 2$. For the offline stage, two system inputs $x_1 = 1$, and $x_2 = 2$ are arbitrarily chosen. Accordingly, the offline simulation data set $\{\boldsymbol{Y}_i\}_i^I$ can be easily obtained as: $\boldsymbol{Y}_1 = \begin{bmatrix} 1 & 2 \\ 3 & 5 \end{bmatrix}$, $\boldsymbol{Y}_2 = \begin{bmatrix} 4 & 8 \\ 10 & 18 \end{bmatrix}$. Suppose there is a sensor at location $d_1 = 0$. The objective is to make a system response prediction for a new input: $x = 1.5$, for which, as a comparison, the true system response (using the given function) is: $\boldsymbol{Y} = \begin{bmatrix} 2.25 & 4.5 \\ 6 & 10.5 \end{bmatrix}$.

3.4.1.1 Application of the Proposed Approach

Step 1: The HOSVD method is applied to decompose the offline simulation data $\chi = \{\boldsymbol{Y}_i\}_i^2 \in \mathbb{R}^{2\times2\times2}$ into eigen spatial and eigen temporal matrices following equation (3.1). To do this, first, the tensor data χ is flattened with respect to the dimension of system input, space and time as: $\chi_{(1)} = \begin{bmatrix} 1 & 2 & 4 & 8 \\ 3 & 5 & 10 & 18 \end{bmatrix}$, $\chi_{(2)} = \begin{bmatrix} 1 & 4 & 3 & 10 \\ 2 & 8 & 5 & 18 \end{bmatrix}$, $\chi_{(3)} = \begin{bmatrix} 1 & 3 & 2 & 5 \\ 4 & 10 & 8 & 18 \end{bmatrix}$. Next, the eigen matrices $\boldsymbol{U}$, $\boldsymbol{V}$ and $\boldsymbol{W}$ are computed from the SVD [75] of the flattened tensor data $\chi_{(1)}$, $\chi_{(2)}$and $\chi_{(3)}$ respectively. For example, the SVD of $\chi_{(1)}$ can be computed, as given in equation (3.10):

$$\chi_{(1)} = \mathbf{U\Lambda D}^T = \begin{bmatrix} -0.40 & 0.92 \\ -0.92 & -0.40 \end{bmatrix} \begin{bmatrix} 23.30 & 0 & 0 & 0 \\ 0 & 0.47 & 0 & 0 \end{bmatrix} \begin{bmatrix} -0.14 & -0.57 & -0.55 & -0.60 \\ -0.23 & -0.30 & -0.50 & 0.78 \\ -0.46 & -0.59 & 0.66 & 0.06 \\ -0.85 & 0.49 & -0.14 & -0.15 \end{bmatrix} \quad (3.10)$$

Meanwhile, based on the singular values, it is noted that the percentage of the information retained by the first eigenvector [75] is, as calculated by equation (3.11):

$$\alpha = \frac{\sum_{i=1}^{r} \sigma_i^2}{\sum_{i=1}^{I} \sigma_i^2} = \frac{23.30^2}{23.30^2 + 0.47^2} = 99.95\% \tag{3.11}$$

as shown in equation (3.11), σ_i is the singular value corresponding to the i-th eigenvector of the eigen matrix $\boldsymbol{U}$. Here, the rank is determined to be 1 for the first dimension which can maintain 99.95% of information. Thus, the eigen matrix can be truncated as $\widetilde{\boldsymbol{U}} = \begin{bmatrix} -0.40 \\ -0.92 \end{bmatrix}$.

Similarly, the truncated eigen spatial and eigen temporal matrices can be calculated for $\chi_{(2)}$ and $\chi_{(3)}$ as $\widetilde{\boldsymbol{V}} = \begin{bmatrix} -0.48 \\ -0.88 \end{bmatrix}$ and $\widetilde{\boldsymbol{W}} = \begin{bmatrix} -0.27 \\ -0.96 \end{bmatrix}$, respectively. Next, by projecting the tensor data χ onto the truncated eigen spatial and eigen temporal matrices, the projected parameter $\mathcal{A}$ (from equation (3.3)) can be obtained as $\mathcal{A} = \chi \times_2 \widetilde{\boldsymbol{V}} \times_2 \widetilde{\boldsymbol{W}} = \begin{bmatrix} 9.21 \\ 21.40 \end{bmatrix}$.

Step 2: The GP model is constructed based on the training data $\{(1, 9.21), (2, 21.40)\}$ obtained from step 1. The estimation of the model parameters is conducted through the maximum likelihood. The trained GP model is used to estimate the variations of parameter matrix $\boldsymbol{A}$ over different input x. Here, the GP model is used to predict the parameter matrix $\boldsymbol{A}$ for the new system input x=1.5. As a result, the parameter matrix $\boldsymbol{A}$ is estimated using a Gaussian distribution with the mean value of 15.30 and standard deviation of 0.25.

The predicted projected parameter is integrated with the eigen spatial and eigen temporal matrices (from equation (3.5)) for spatio-temporal prediction, as in equation (3.12):

$$\hat{\mathbf{Y}}_{k=k_0} = 15.30\begin{bmatrix}-0.27\\-0.96\end{bmatrix}\begin{bmatrix}-0.48 & -0.88\end{bmatrix} = \begin{bmatrix}1.97 & 3.59\\7.10 & 12.92\end{bmatrix}. \tag{3.12}$$

Thus, an initial prediction of the spatio-temporal response is obtained.

Step 3: The initial prediction of parameter, as well as the system response is treated as prior pdf, which is to be updated in this step. The prior pdf of parameter $\boldsymbol{A}$ is discretely represented by weighted particles. Here, for simplicity, only 5 particles are used. In reality, the number of particles is much more. Initially, the particles (or samples) are randomly generated from the estimated Gaussian distribution $N(15.30, 0.25^2)$ as {16.74, 15.63, 14.55, 16.67, 13.59}, which are generated using the Matlab's 'randn' function. These particles are assigned with equal weights as 0.2 initially.

Then, with the collection of the sensor measurement data at location d_1, the weight of each of the particle is updated. Assume the sensor measurement at time step $k_1 = 1$ is 2.2 with a white noise standard deviation of 0.1. To update the weight of the first particle {16.74}, the likelihood of the first particle given the sensor measurement is calculated (from equation (3.8)) as:

$$\begin{gathered}\hat{\mathbf{Y}}^1_{k=k_0} = 16.74\begin{bmatrix}-0.27\\-0.96\end{bmatrix}\begin{bmatrix}-0.48 & -0.88\end{bmatrix} = \begin{bmatrix}2.16 & 3.93\\7.77 & 14.14\end{bmatrix}\\ P\left(s_{1,1}\middle|\mathbf{A}^1_0\right) = P\left(s_{1,1} - y^1_{1,1}\middle|R^1_1\right) = P\left((2.2-2.16)\middle|0.1\right) = 3.65\end{gathered} \tag{3.13}$$

Similarly, the likelihood of the other four particles can be computed as {0.72, 0.02, 3.50, 0.00}. Then the weight of the first particle can be updated (from equation (3.9)) as:

$$w_1^1 = \frac{w_0^1 P\left(s_1 \middle| \mathbf{A}_0^1\right)}{\sum_{p=1}^{5} w_0^p P\left(s_1 \middle| \mathbf{A}_0^p\right)} = \frac{3.65}{3.65 + 0.72 + 0.02 + 3.50 + 0.00} \approx 0.46 \tag{3.14}$$

Following the same procedure, the weight for each particle can be updated. The estimated mean of the parameter at time step $k_1 = 1$ can be obtained as:

$$\mathbf{A}_1 = \sum_{p=1}^{5} w_1^p \mathbf{A}_0^p \approx 16.69\,, \tag{3.15}$$

as well as the spatio-temporal prediction can be updated as:

$$\hat{\mathbf{Y}}_{k=k_1} = 16.69 \begin{bmatrix} -0.27 \\ -0.96 \end{bmatrix} \begin{bmatrix} -0.48 & -0.88 \end{bmatrix} = \begin{bmatrix} 2.15 & 3.92 \\ 7.74 & 14.09 \end{bmatrix} \tag{3.16}$$

To evaluate the prediction performance, the absolute error is calculated as the absolute difference between the predicted spatio-temporal response and true response. Comparing with the true response values mentioned at the beginning, the prediction at location $d_1 = 0$ for two time steps has dropped from 0.28 and 0.91 to 0.1 and 0.58 respectively. Note that the proposed approach assumes the system responses at different locations are correlated, since only parital sensor measurements are used to update the system reponse at all locations. If the system responses are not correlated, for the location without sensor measurements, e.g., the location d_1 in this case study, the prediction error might not be greatly reduced during the data assimilation process. Additionaly, this numerical example is mainly used to demonstrate the application of the proposed approach. For better accuracy, the number of particle should be much more than 5 in order to obtain a response prediction with comparable acccuracy to that obtained from the simulation model.

3.4.2 Aeroelastic Response Prediction

This engineering example was considered in a previous work [71] and Chapter 2. The goal for the example is to estimate the nonlinear aeroelastic responses, including flutter behavior, of a joined-wing aircraft called SensorCraft [1], for varying flight conditions. In this case study, the simulation model for estimating the nonlinear aeroelastic behavior during the offline stage is executed through a co-simulation strategy [47, 50, 76]. The aerodynamic mesh of the joined-wing aircraft is provided elsewhere [76]. The beam model of the left-wing structure of the aircraft is given in Figure 3.3.

The system inputs $\boldsymbol{x}$ for the aeroelastic simulator are: airspeed, angle of attack, and atmospheric air density. The variations in inputs are assumed to be representative of different flight conditions during the online operation. The outputs of the simulator are the time-varying nodal displacements at 19 discrete locations on the left wing of the aircraft, where the locations are indicated as $\{\mathbf{d}_j\}_{j=1}^{19}$ in Figure 3. In this example, only the displacements along the vertical direction are considered for demonstrating the proposed approach.

To study the aeroelastic stability, the simulation time was set for a sufficiently high number of time steps (74,882 time steps for each flight condition) so that the initial transient behavior dies out. The initial transient effects persist for approximately 10,000 time steps, then the ensuing phase is a steady-state oscillation. The proposed approach has been implemented to estimate both the transient and steady-state behaviors. Since the author mainly concentrates on determining the presence of flutter instability, the

time interval of 1,000 time steps (approximately 10 periods of oscillation) within the steady state response history is selected for demonstration.

In this example, three sensors are used, and the spatial locations of each sensor are arbitrarily selected from the 19 simulated locations. As indicated by the green circles in Figure 3.3, the first sensor is chosen to be located on the left inward forward wing, the second sensor is placed at the left rear wing, and the third sensor is situated at the left outward forward wing. Here, the sensor data are synthetic, generated from the simulated model results by adding Gaussian noise with zero mean and standard deviation of 0.04 for a nondimensional displacement (6% of the average amplitude). However, no prior knowledge of the sensor noise is used in the application of the proposed approach.

In this engineering example, the aircraft's aeroelastic behavior at nine different flight conditions within the post-flutter regime (after the onset of flutter) is evaluated by using the aeroelastic simulator. The combinations of nine different flight conditions are shown in Table 3.2. For these nine cases, they are set to have the same air density. This is done because the objective in this example is to interpolate the nonlinear aeroelastic response and possibly identify the flutter boundary (e.g., onset of the flutter speed) for a fixed air density. However, the approach can also be used for prediction using any other air density value. Here, a nine-fold cross-validation is used to assess the predictive ability of the proposed approach. That is, each of these nine flight conditions is used to simulate the proposed online prediction approach during the aircraft operation.

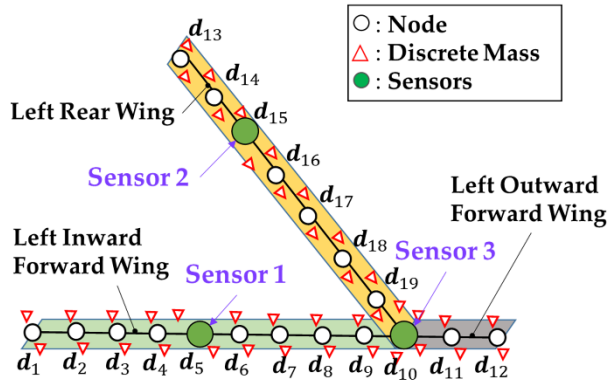


Figure 3.3 Beam representation of left-wing structure of the joined-wing SensorCraft

Table 3.2 Nine flight conditions used in aeroelastic simulations

Flight operational conditions $\boldsymbol{x}$	Airspeed $v(m/s)$	Angle of Attack $AoA(^\circ)$	Air Density $\rho(kg/m^3)$
1	160	5	0.1152
2	161	5	0.1152
3	162	5	0.1152
4	163	5	0.1152
5	164	5	0.1152
6	165	5	0.1152
7	170	10	0.1152
8	175	10	0.1152
9	180	5	0.1152

3.4.2.1 Application of the Proposed Approach

For this example, the prediction procedure is only detailed, step-by-step, for one of the flight conditions:$v = 165m/s$, $AoA = 5^\circ$, and $\rho = 0.1152kg/m^3$. The prediction results for all of the remaining cases are summarized in the next section.

In the first step of the proposed approach, HOSVD is used to obtain the eigen spatial and eigen temporal matrices from the offline simulation data. Then, these matrices are truncated based on the accumulative percentage of eigenvalue in order to preserve over 98% of spatio-temporal information. Accordingly, the number of eigenvectors in the spatial and temporal dimensions is chosen as 2 and 6, respectively. Subsequently, a low-dimensional projected parameter tensor $\mathcal{A} \in \mathbb{R}^{8\times6\times2}$ can be obtained by using

equation (3.3). The projected parameter tensor governs the interaction between eigen spatial and eigen temporal matrices to help with the prediction of the aircraft's spatio-temporal behavior.

The second step is to construct the GP models to characterize nodal displacements through modeling of the low-dimensional projected parameters, which are trained by the projected parameter tensor $\mathcal{A}$ obtained in the first step. Here, each slice of the projected parameter tensor $\boldsymbol{A} \in \mathbb{R}^{6\times 2}$, composed of 12 elements of parameters, is a representation of the aircraft's spatio-temporal behavior corresponding to each flight condition. These parameter elements are independently modeled and thus 12 GP models are constructed. The 12 GP models, together with the eigen spatial and eigen temporal matrices obtained in the first step, are obtained offline for subsequent online usage. They can be used as an initial estimate (prior) of the system behavior during online operation, as indicated in equation (3.5).

The preliminary prediction of the time-varying nodal displacement, namely at $k = k_1$, is depicted in Figure 3.4. In Figure 3.4(a), the contour plot of the simulated nodal displacements at 19 locations for 1,000 time steps by the prescribed aeroelastic simulator is shown. The prior predictions computed from the offline models are illustrated in Figure 3.4(b). Meanwhile, the absolute prediction errors between the aeroelastic simulator and the initial estimate are demonstrated in Figure 3.4(c). It can be seen that the initial predictions provide reasonably accurate estimates of nodal displacements for majority of the spatial locations, including at locations $\boldsymbol{d}_1$ to $\boldsymbol{d}_{13}$. Only small magnitude and phase differences of the nodal displacements are notable at these locations. However, the estimates from locations $\boldsymbol{d}_{14}$ to $\boldsymbol{d}_{19}$ at the aircraft's left

rear wing, have noticeable prediction errors mainly due to the magnitude and phase differences.

The third step is to recursively update the model estimates with the collection of sparse sensor measurements. This is achieved via the extended ROPF, and the three representative results obtained are shown in Figure 3.5. In this figure, the author shows the contour plot of the absolute prediction errors at the 5th, 100th, and 200th assimilation cycles separately. After five assimilation iterations, as displayed in Figure 3.5(a), for the first 200 time steps the prediction error has largely reduced amongst all the spatial locations, in contrast to the prior prediction error in Figure 3.4(c). These predictions still decay as they extend further into the future time steps. For instance, as the assimilation process arrives at the 100th time step, an additional reduction of prediction errors for the first 400 time steps can be obtained, as shown in Figure 3.5(b). However, there still exist discernible errors at locations $\boldsymbol{d}_{14}$ and $\boldsymbol{d}_{19}$ Eventually, the results for reaching the 200th assimilation iterations are shown in Figure 3.5(c). The prediction errors for all the spatial locations as well as the subsequent 800-time steps have decreased significantly.

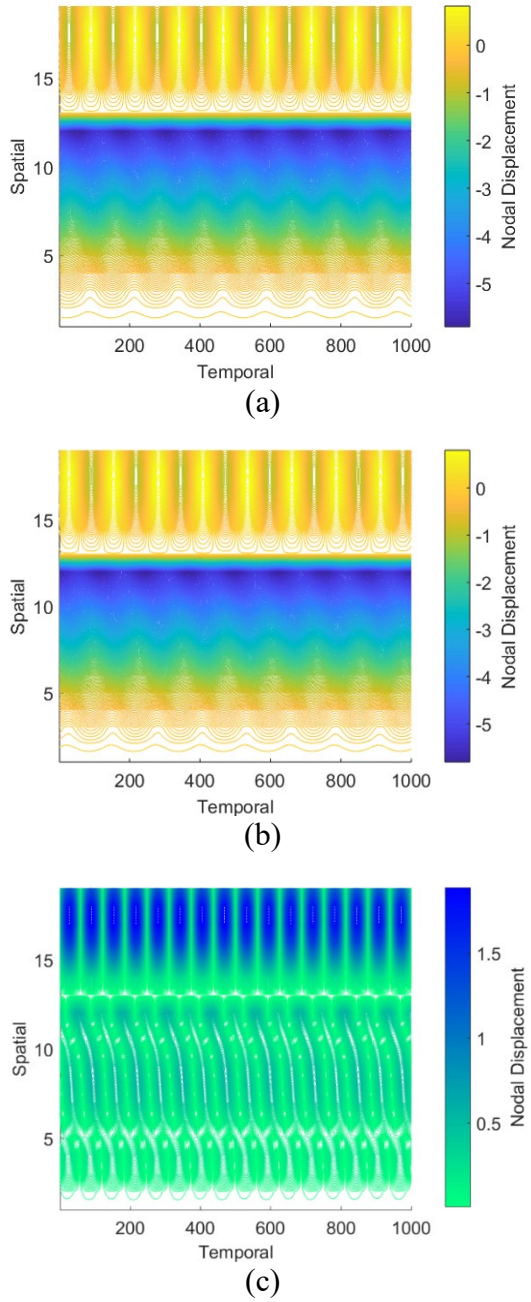


Figure 3.4 Initial prediction at $k = k_1$, (a) simulated system behavior, (b) offline model prediction, and (c) predicted absolute error

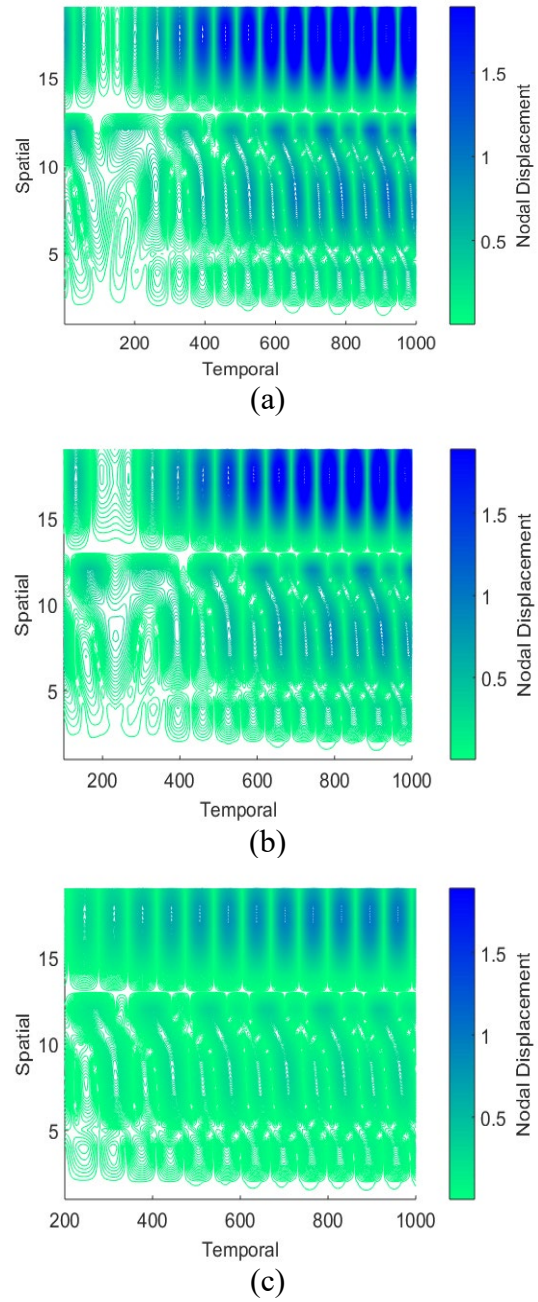


Figure 3.5 Online data assimilation process: the predicted absolute errors at (a) $k = k_5$ (b) $k = k_{100}$, and (c) $k = k_{200}$

3.4.2.2 Results and Discussions

The final prediction results for each of the nine flight conditions are provided in Table 3.3. These results illustrate the average absolute prediction errors, as well as the

relative errors with respect to the nonlinear aeroelastic simulator during the online prediction. The average absolute error is calculated as the average absolute difference between the high-fidelity simulation data and predicted results. The relative error is obtained by dividing the average absolute error by the average absolute of simulated responses. Through the proposed approach, the author has demonstrated the predictive capability comparable to the high-fidelity simulation, where all the relative errors are within 5%. For the predictions in cases 6-9, the preliminary estimates provided from the offline models can attain equitable precision, and assimilation procedure converges to the simulation level accuracy rather quickly. However, in flight conditions 1-5, these five cases have a less ideal fit initially and necessitate longer assimilation cycles for convergence. These cases are near the flutter boundary (onset flutter speed of $156m/s$), and the exceptionally distinct displacement modes are involved in these datasets. The obtained eigen spatial and eigen temporal matrices can be deficient in characterizing the inherent features in these cases. Meanwhile, similar to the state estimate illustrated in Figure 3.4(c), the preliminary prediction errors mostly exist in the left rear wing (locations between $\boldsymbol{d}_{14}$ and $\boldsymbol{d}_{19}$) due to the phase and magnitude differences. However, with the sparse sensor measurements gathered from the aircraft's different regions, the proposed approach can gradually capture these distinct displacement modes, and reduce the prediction errors in the rear wing through utilization of the spatio-temporal dependencies. Additionally, the number of simulated datasets in this case study is limited. With an increased number of training datasets, there is a higher chance to have scenarios with similar oscillation modes. This would further enhance the model's predictive ability.

To further study the proposed approach, the effect of sensor placement on the prediction performance has been investigated. The three sensor locations, $\boldsymbol{d}_5$, $\boldsymbol{d}_{10}$, and $\boldsymbol{d}_{15}$, are initially arbitrarily selected out of the 19 positions illustrated in Figure 3.3. Next, 100 design scenarios for sensor locations are randomly generated to assess the sensitivity of the prediction with respect to the placement of sensors. Following the application of the proposed approach, the average absolute error of the prediction for each of the sensor location configurations is obtained. The representative results are shown in Table 3.4 and Table 3.5. In Table 3.4, the first 10 configurations of sensor locations with the lowest absolute error are displayed. It is interesting to note that these sensor placements scenarios share a similar pattern. They are evenly distributed at three regions, as indicated by different colors in Figure 3.3, where green stands for the left inward forward wing, yellow for the left rear wing, and grey for the left outward forward wing. On the other hand, the 10 configurations for the sensor locations producing the highest prediction errors are presented in Table 3.5. In these cases, they have not covered each of the marked region. The author believes the reason for the phenomenon is that these three regions contain dynamical behaviors distinct from one another. In the cases that the sensors are evenly distributed amongst each of the three regions, the proposed approach has shown good performance for different sensor location combinations as shown in Table 3.4. While for the cases that the sensors are closely located in the same region, as illustrated in Table 3.5, a relatively larger prediction error is obtained. One explanation can be that the model prediction tends to be overfitted to the system behavior of one region, where sensors are closely located, during the data assimilation process. However, in reality, the sensors are less likely to

be placed closely in one region, especially when only a limited number of sensors is used. Therefore, the proposed prediction appears to be robust to sensors' spatial locations for this example.

Table 3.3 Average absolute and relative prediction errors for 9 flight conditions

Conditions	1	2	3	4	5
Abs. error	0.059	0.056	0.064	0.058	0.059
Rel. error	4.37%	4.12%	4.64%	4.13%	4.14%
Conditions	6	7	8	9	Mean
Abs. error	0.051	0.049	0.052	0.054	0.056
Rel. error	3.53%	2.11%	2.15%	3.21%	3.60%

Table 3.4 Ten sensor placement configurations with the lowest absolute errors

Location 1	Location 2	Location 3	Mean abs. error
$\mathbf{d}_8$	$\mathbf{d}_{12}$	$\mathbf{d}_{15}$	0.053
$\mathbf{d}_6$	$\mathbf{d}_{12}$	$\mathbf{d}_{19}$	0.054
$\mathbf{d}_9$	$\mathbf{d}_{11}$	$\mathbf{d}_{15}$	0.058
$\mathbf{d}_5$	$\mathbf{d}_{11}$	$\mathbf{d}_{15}$	0.058
$\mathbf{d}_8$	$\mathbf{d}_{11}$	$\mathbf{d}_{17}$	0.058
$\mathbf{d}_6$	$\mathbf{d}_{10}$	$\mathbf{d}_{17}$	0.059
$\mathbf{d}_7$	$\mathbf{d}_{10}$	$\mathbf{d}_{17}$	0.059
$\mathbf{d}_9$	$\mathbf{d}_{12}$	$\mathbf{d}_{18}$	0.062
$\mathbf{d}_5$	$\mathbf{d}_{10}$	$\mathbf{d}_{14}$	0.065
$\mathbf{d}_4$	$\mathbf{d}_{10}$	$\mathbf{d}_{14}$	0.067

Table 3.5 Ten sensor placement configurations with the highest absolute errors

Location 1	Location 2	Location 3	Mean abs. error
$\mathbf{d}_5$	$\mathbf{d}_{17}$	$\mathbf{d}_{19}$	0.430
$\mathbf{d}_1$	$\mathbf{d}_7$	$\mathbf{d}_{12}$	0.445
$\mathbf{d}_{10}$	$\mathbf{d}_{13}$	$\mathbf{d}_{16}$	0.453
$\mathbf{d}_7$	$\mathbf{d}_{18}$	$\mathbf{d}_{19}$	0.461
$\mathbf{d}_3$	$\mathbf{d}_{13}$	$\mathbf{d}_{18}$	0.516
$\mathbf{d}_8$	$\mathbf{d}_{15}$	$\mathbf{d}_{16}$	0.520
$\mathbf{d}_2$	$\mathbf{d}_5$	$\mathbf{d}_{12}$	0.557
$\mathbf{d}_1$	$\mathbf{d}_2$	$\mathbf{d}_{18}$	0.610
$\mathbf{d}_{15}$	$\mathbf{d}_{16}$	$\mathbf{d}_{18}$	0.620
$\mathbf{d}_{13}$	$\mathbf{d}_{16}$	$\mathbf{d}_{18}$	0.666

In terms of wall-clock time, the offline model data of this case study can be used to make an initial prediction of the spatio-temporal system behavior within 0.01 seconds. Then, for the proposed data assimilation processes, each iteration takes about 0.05 seconds. In total, to assimilate sensor measurements for 1,000-time steps, the proposed approach takes approximately 50 seconds to complete the entire prediction process. However, as shown in the example, the predictions converge sooner, in around 200 assimilation iterations (i.e., about 10 seconds of clock time). The computing platform used for this case study was a Dell computer running Windows 10, with Intel Xeon CPU E3-1245 v5 3.50GHz and 16.0 GB RAM.

3.4.3 Robotic Appendage Forces Prediction

In this subsection, the effectiveness of the data-driven approach proposed from this chapter is demonstrated by predicting the behavior of a rigid robotic appendage as it interacts during its motion on a terrain with granular material [77]. The motion of this rigid body is prescribed to be consistent with the limited physical experimental data reported in the literature for a prototype of the robot [78]. The goal here is to predict the drag and lift forces for the robotic appendage as a function of a time varying quantity during the motion. Here, the time varying quantity is the angle between the appendage and vertical direction, as the appendage moves along a clockwise direction through the granular material. For this example, the robotic appendage design and operation can be specified by its length, width, depth, curvature, and a clockwise angular motion stride frequency, respectively. As shown in Figure 3.6(a)-(e) [77], a robotic appendage design can be a reversed C-leg, reversed L-leg, flat leg, L-leg, and C-leg, with their corresponding real-life examples of different leg morphologies.

In this case study, two sources of data are available: one is from a multi-response physics-based simulation model representing the interacting forcing behavior (drag and lift forces) of the appendage, and the other is from a limited number of data obtained by physical experiments. A high-fidelity physics-based simulation [77] is used to collect a set of data for different robotic leg designs and operational conditions, the $\boldsymbol{x}$ values. Next, seven cases of leg designs and operations are studied, as illustrated in Table 1, where the physical parameters for each leg are obtained from the literature [77]. These robotic legs interact with the terrain during the walking motion. The terrain with the granular media is closely packed with 3 mm (diameter) glass spheres having a particle material density of 2.6 g/cm^3 and a repose angle of 23°. During the robotic leg's motion on the terrain, it is assumed that the leg is first paused for 2 seconds, and then rotates clockwise at a hip height of 2 cm within the leg angle range of: $-3\pi/4 \leq \theta \leq 3\pi/4$. For a combination of robot leg designs and operations, high-fidelity simulations are conducted to collect data for the leg's drag and lift forces [77]. On the other hand, the data from physical experiments [78] are only available for three subsets of the leg designs and operations, which are the flat leg, C-leg, and reverse C-leg with a stride frequency of 0.2 rad/s.

For this example, first, the proposed prediction approach is used to predict the behavior for the flat robot leg design using several operational conditions, as provided in subsection 3.4.3.1. Then, the results for all robot leg designs and discussions will be presented in subsection 3.4.3.2.

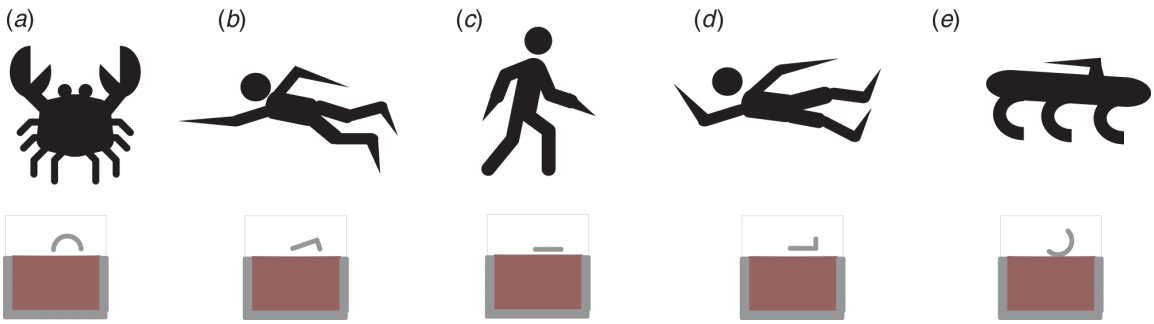


Figure 3.6 The figures in the top row are sketches of real-life examples of different leg morphologies. The figures in the bottom row are for corresponding leg appendages considered: (a) crab, reversed C-leg, (b) freestyle, reversed L-leg, (c) human walking, flat leg, (d) backstroke, L-leg, and (e) C-leg robot, C-leg. Courtesy of [77]

Table 3.6 Mixture of robotic leg design and operational scenarios for simulation data

Robotic appendage design	Operation condition
Flat leg: fixed leg length	Varying stride frequency
C-leg: fixed leg radius	Varying stride frequency
Reversed C-leg: fixed leg radius	Varying stride frequency
L-leg: fixed leg length	Varying stride frequency
Reversed L-leg: fixed leg length	Varying stride frequency
L-leg: varying foot length	Fixed stride frequency
Reversed L-leg: varying foot length	Fixed stride frequency

3.4.3.1 Application of Prediction Approach to Flat Robot Leg Design

The flat robotic leg design and its real-life leg morphology can be visualized in Figure 3.6(c), where the operating profiles for the leg at different representative times can be also gleaned in Figure 3.10(a). As shown in the first row of Table 3.6, the simulation is conducted to generate data for the flat leg design with fixed leg length but with varying operating conditions (by changing the stride frequency), where 10 nondimensionalized stride frequencies ranging from 1 to 10 are considered.

For testing the proposed data-driven prediction framework, a ten-fold cross-validation algorithm is used. That is, each of the ten operating conditions is used as a testing data, while the other nine conditions are used as training data sets. The

prediction approach is then used to predict the lift and drag forces for each of the ten operational conditions (stride frequencies). In this subsection, an application of the proposed approach is only detailed for one of the stride frequencies, i.e., 0.6 rad/s (or nondimensionalized stride frequency of 3). In the next subsection, the prediction results for all the remaining cases are summarized.

In terms of the detailed description for that one stride frequency, first, HOSVD is used to decompose the training data sets into eigen operation and temporal-like matrices (see eigen spatial and temporal matrices discussed earlier), respectively. Then, these matrices are truncated based on the accumulative percentage of eigenvalue to preserve over 95% of the operational and temporal-like information. Accordingly, the number of eigenvectors in the operational and temporal-like dimensions is chosen as 4 and 7, respectively. Subsequently, a low-dimensional projected parameter can be obtained by using equation (3.3), wherein the value of the project parameters corresponding to each operational condition (or nondimensionalized stride frequency) are plotted, as shown with a green plus symbol in Figure 3.7. Second, GP models are constructed to model the variation of low-dimensional parameters along operational conditions, as shown in Figure 3.7. The GP model predictions and its 95% prediction confidence intervals are, respectively, visualized as blue lines and in-between shaded grey areas in Figure 3.7. Then, the GP models are used to predict the parameters for the tested operational condition, the nondimensionalized stride frequency of 3. The predicted values are indicated by the red triangles in Figure 3.7. With the integration of the predicted parameters and eigen operation and temporal-like matrices, the prediction

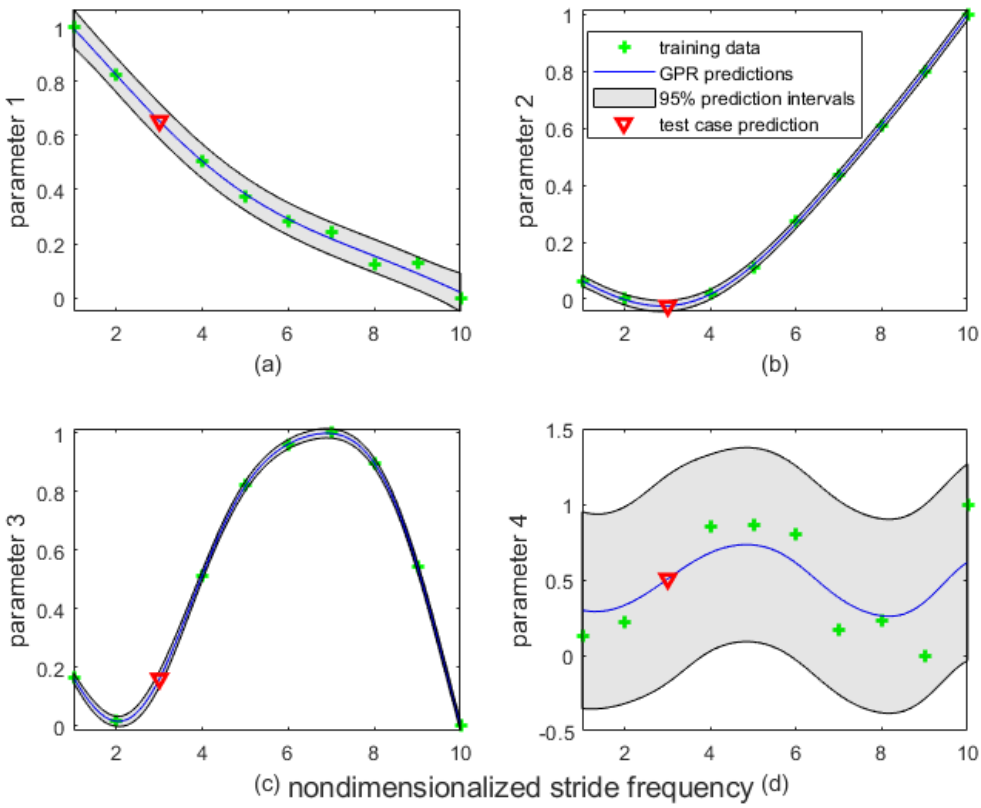


Figure 3.7 Gaussian process models prediction for four low dimensional parameters at the test operational condition, stride frequency 0.6 rad/s (nondimensionalized 3)

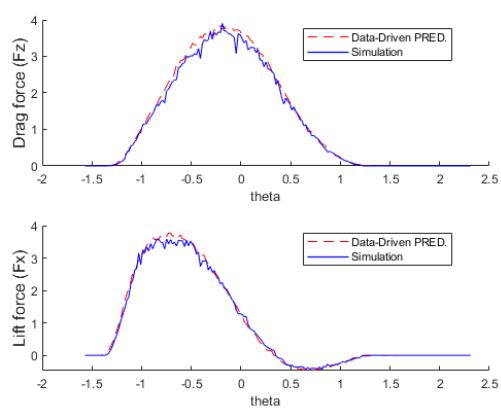


Figure 3.8 Comparison of drag and lift forces between simulation and prediction approach (noted as Data-Driven PRED)

of the drag and lift forces can be obtained, as shown in Figure 3.8, which has a very comparable accuracy to the results obtained from the high-fidelity simulation.

A similar application of the prediction approach is applied to predict the drag and lift forces for each of the 10 operational conditions. In Figure 3.9, a few snapshots of the robotic leg during its interaction with the terrain is shown in Figure 3.9 (a1). The predicted drag and lift forces with nondimensionalized stride frequencies ranging from 1 to 10 are illustrated in Figure 3.9 (a2) and (a3), respectively. Similarly, the blue solid lines are for simulated results, and the red dashed lines are for the predicted results. The results show that the data-driven prediction approach can generate predictions that have comparable accuracy as simulations across different stride frequencies, while also significantly reduce the computational effort.

On the other hand, sparse physical experimental data [78] is available for the tested operational condition, with the stride frequency of 0.2 rad/s (nondimensionalized as 1). So, the ROPF approach has been used to further update the prediction by using assimilation with the sparse physical experiment data. Comparison of the drag and lift forces have been provided in Figure 3.9(a4), which includes the physical experimental data, simulation data, and the proposed prediction with and without the ROPF approach. As can be gleaned in Figure 3.9(a4), the prediction matches well with both simulation and experimental results, and outperforms the simulation by being closer to the physical experiments. Additionally, the average absolute errors between the simulation and prediction for all 10 nondimensional stride frequencies (1 to 10) are calculated as shown in Figure 3.9(a6).

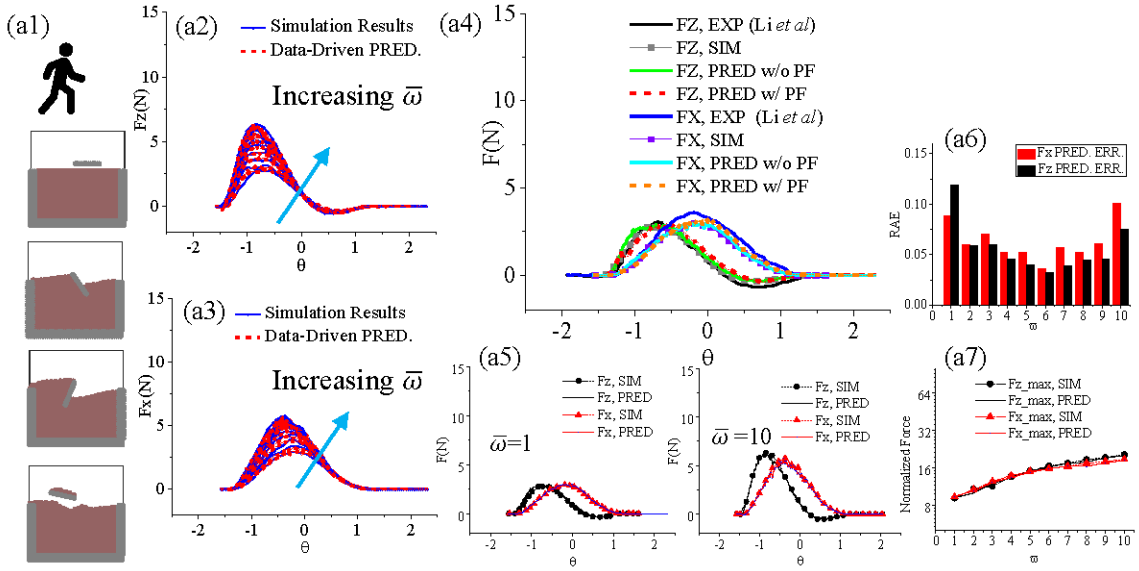


Figure 3.9 Comparison of data driven prediction and physics-based simulation for flat leg, (a1) Real life leg morphology and profiles at different representative time, (a2) Net lift Fz, (a3) Net thrust Fx, (a4) Comparison of net forces from experiments, simulations, data-driven prediction with particle filters, data-driven prediction without particle filters, (a5) Net forces for nondimensionalized frequency of $\boldsymbol{\omega}$ =1 and $\boldsymbol{\omega}$ =10 , (a6) Relative absolute error for net forces prediction, (a7) Sensitivity dependence of normalized net forces on $\boldsymbol{\omega}$.

3.4.3.2 Results and Discussions

Following the same application, the proposed data-driven approach has been evaluated on a combination of robotic appendage designs and operational conditions from Table 3.6. That is, with the fixed leg radius of C-leg design (as shown in Figure 3.6(e)), the prediction approach is followed to predict the lift and drag forces for varying stride frequencies (nondimensionalized 1 to 10) when the robotic appendage moves through the granular material. The predicted results comparing with high-fidelity simulation are shown in Figure 3.10(a1)-(a7). Similarly, the predicted results with the fixed leg radius of reverse C-leg design (as shown in Figure 3.6(e)) and with varying stride frequencies are visualized in Figure 3.10(b1)-(b7). The comparison study between data-driven approach and simulation is performed for a fixed leg length of L-leg design and varying stride frequencies are illustrated in Figure 3.11(a1)-(a6); and for

the fixed leg length of reverse L-leg design and varying stride frequencies are plotted in Figure 3.11(b1)-(b6).

Additionally, the comparison of data-driven predictions and simulation results is extended to two other cases. One case is for the fixed operational condition (stride frequency), the leg length of L-leg design is varying from 0.1 to 1 meter, where the predicted interaction forces for different leg length are shown in Figure 3.12(a1)-(a5). Another case is for the fixed operational condition (stride frequency), the leg length of reverse L-leg design is varying from 0.1 to 1 meter, as plotted in Figure 3.12(b1)-(b5).

From these results, it is shown that data-driven prediction technique presented in this chapter can generate predictions that have comparable accuracy as the simulations. It can capture robotic appendage interaction forces for both varying operational conditions and leg length designs. With both simulation data and sparse experimental measurement as input, the data-driven approach embedded ROPF techniques could have the potential to outperforming simulations in the long-term predictions.

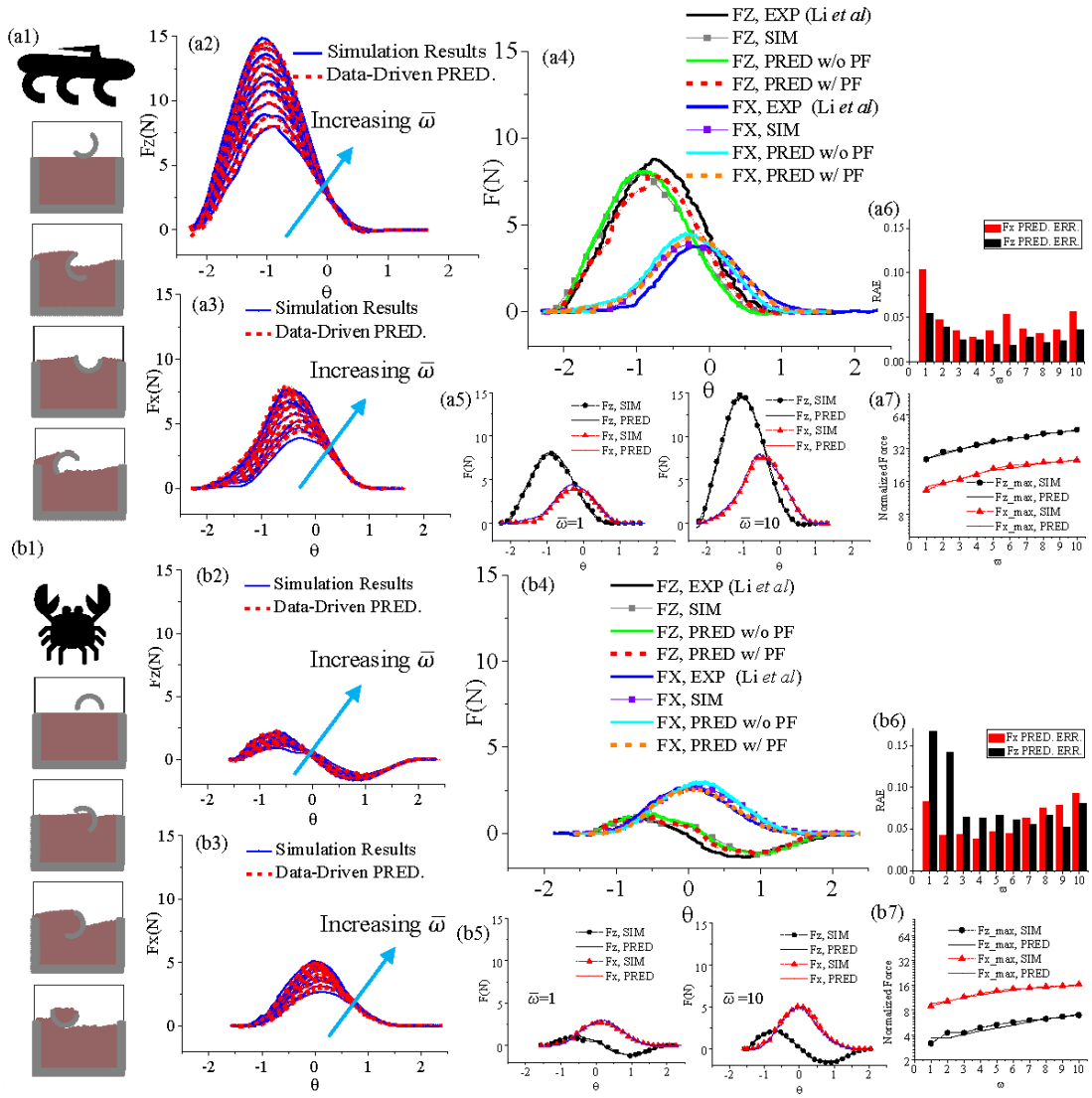


Figure 3.10 Comparison of data driven prediction and physics-based simulation for C-leg and Reversed C-leg across different rotation speeds. a) C-leg: (a1) Real life leg morphology and profiles at different representative time, (a2) Net lift Fz, (a3) Net thrust Fx, (a4) Comparison of net forces from experiments, simulations, data-driven prediction with particle filters, data-driven prediction without particle filters, (a5) Net forces for $\boldsymbol{\omega}$ =1, and $\boldsymbol{\omega}$ =10, (a6) Relative absolute error for net forces prediction, (a7) Sensitive dependence of normalized net forces on $\boldsymbol{\omega}$. b) Reversed C- leg: (b1) Real life leg morphology and profiles at different representative time, (b2) Net lift Fz, (b3) Net thrust Fx, (b4) Comparison of net forces from experiments, simulations, data-driven prediction with particle filters, data-driven prediction without particle filters, (b5) Net forces for $\boldsymbol{\omega}$ =1, and $\boldsymbol{\omega}$ =10, (b6) Relative absolute error for net forces prediction, (b7) Sensitive dependence of normalized net forces on $\boldsymbol{\omega}$.

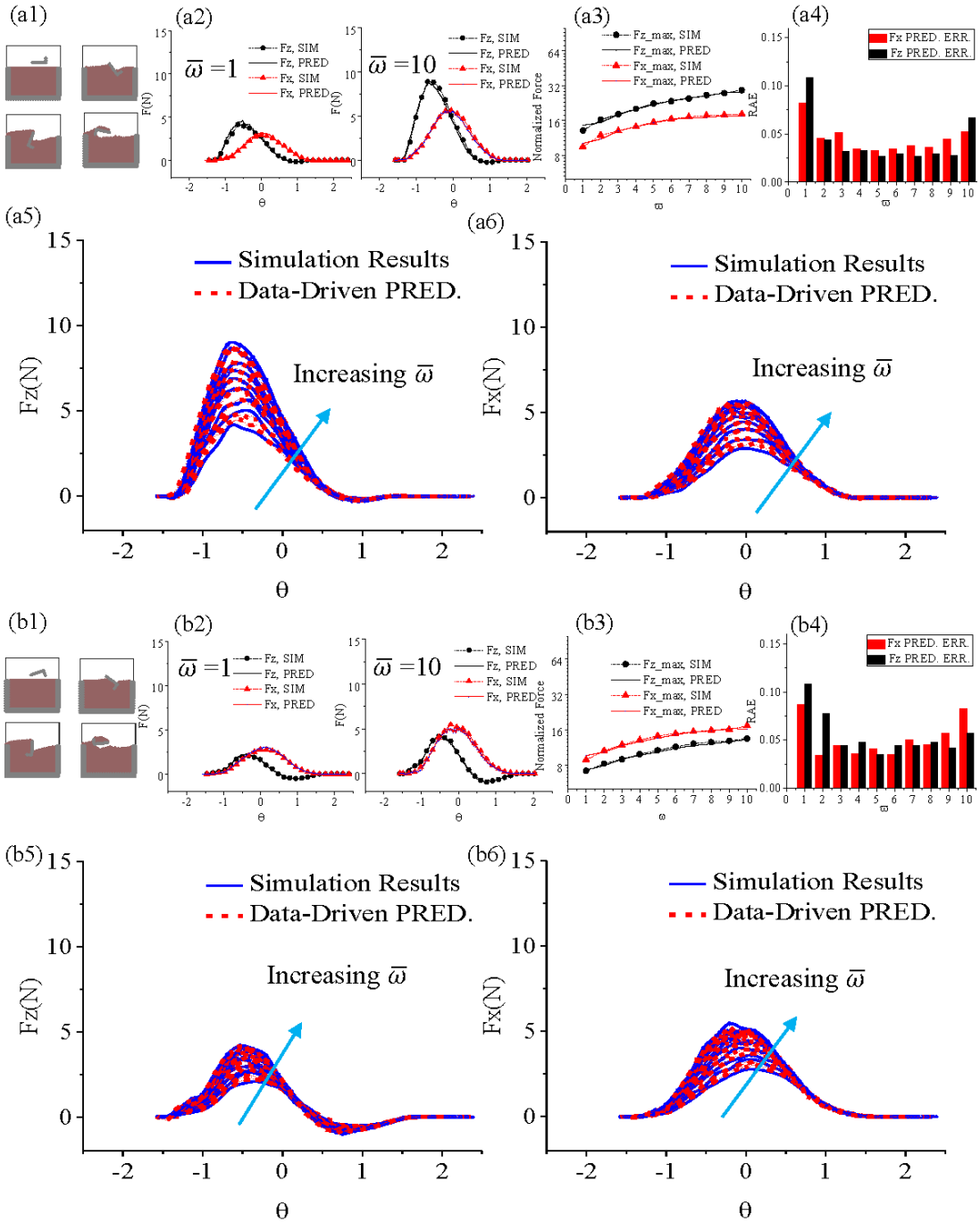


Figure 3.11 Comparison of data driven prediction and physics-based simulation for L-leg and Reversed L-leg across different rotation speeds. a) L-leg, fl=1/3: (a1) Profiles at different representative time, (a2) Net forces for $\boldsymbol{\omega}$ =1, and $\boldsymbol{\omega}$ =10, (a3) Sensitive dependence of normalized net forces on $\boldsymbol{\omega}$ (a4) Relative absolute error for net forces prediction, (a5) Net lift Fz, (a6) Net thrust Fx. b) Reversed L-leg, fl=1/3: (b1) Profiles at different representative time, (b2) Net forces for $\boldsymbol{\omega}$ =1, and $\boldsymbol{\omega}$ =10, (b3) Sensitive dependence of normalized net forces on $\boldsymbol{\omega}$ (b4) Relative absolute error for net forces prediction, (b5) Net lift Fz, (b6) Net thrust Fx.

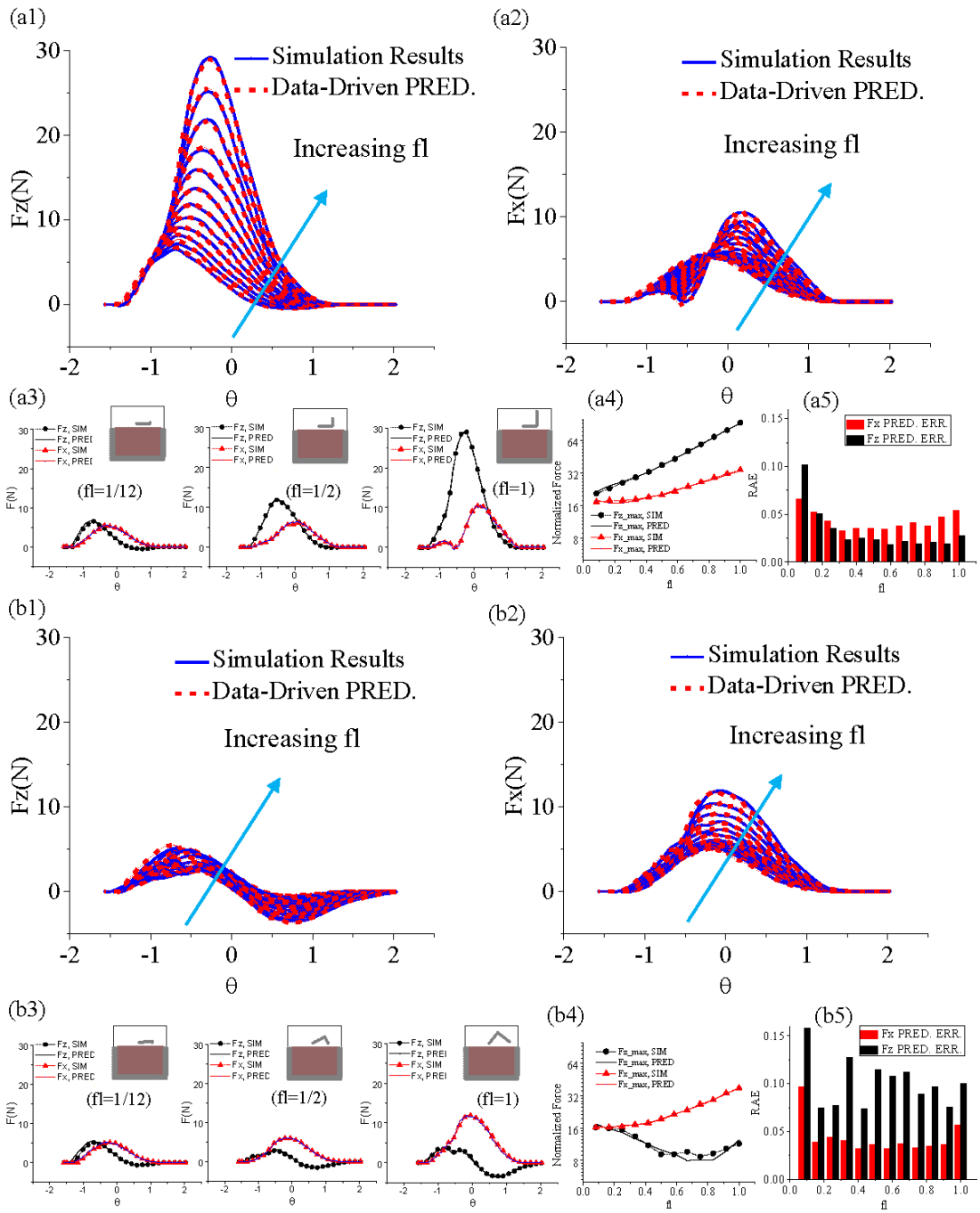


Figure 3.12 Comparison of data driven prediction and physics-based simulation for L-leg and Reversed L-leg across different foot lengths. a) L-leg: (a1) Net lift Fz, (a2) Net thrust Fx, (a3) Net forces for fl=1/12, fl=1/2 and fl=1, (a4) Sensitive dependence of normalized net forces on fl (a5) Relative absolute error for net forces prediction. b) Reversed L-leg: (b1) Net lift Fz, (b2) Net thrust Fx, (b3) Net forces for fl=1/12, fl=1/2 and fl=1, (b4) Sensitive dependence of normalized net forces on fl (b5) Relative absolute error for net forces prediction.

3.5 Summary

In this chapter, a new data-driven approach for prediction of multi-response system behavior is presented. This is done through an integration of the simulation data (offline) and sparse sensors measurement data or sparse physical experiment data. This approach is designed to address three major challenges: 1) predictive modeling of complicated multi-response spatio-temporal dependencies; 2) dealing with sparse, noisy sensors/experiments measurement data; and 3) reducing the computational expense. In this approach, HOSVD is used to reduce the dimension of the simulated data while retaining the spatio-temporal correlations. Next, the GPs integrated with HOSVD are used to capture the system's nonlinear dynamic behavior. Finally, ROPF is extended to assimilate sensor measurements to effectively enhance model predictions, thus capable of dealing with sparse and noisy measurements. The proposed approach is demonstrated through a simple numerical example and two complicated engineering examples. One of these examples is on predicting multi-response aeroelastic behavior for a joined wing aircraft, and the other example on predicting robotic appendage motion. In the first example, with the results obtained for the joined-wing aircraft case study, it is shown that the proposed approach can predict the aeroelastic responses with comparable accuracy to that from the high-fidelity simulations, while significantly reducing the computational cost. Moreover, it is shown the predictions are relatively robust to sensor placement based on the results obtained. In the second example, the robotic appendage example, it is shown that the proposed approach not only has comparable accuracy to high-fidelity simulation but can outperform the simulation after fusing the data from the simulation with those obtained

from the sparse physical experiment data. Moreover, the goodness of the data-driven approach is shown for changing the design and operational conditions in predicting the robot appendage motion.

There might be several advantages in using the proposed approach, when compared with state-of-the-art algorithms in the literature, e.g., Long Short-Term Memory (LSTM) [58]. LSTM has been reported to be not very reliable in handling the effect of data uncertainty [62, 63], but for the proposed approach, the reduced order particle filter shows the capability of dealing with noisy data distribution which can take any form (Gaussian or non-Gaussian). Moreover, LSTM usually requires a large training data to reach a good prediction performance [63]. However, as shown in the case study (Section 3.4.2), both the offline simulation data and online sensor (or experimental) measurements can be limited. For instance, simulating aeroelastic behaviour for each of the flight condition, as shown in Table 3.2, takes more than 10 hours of computational time to obtain. On the other hand, online measurements are obtained by sensors which are sparsely located. Practically speaking, sensor measurements during flutter provide very limited information, because conducting experiments within post-flutter regime is dangerous, and can lead to the premature failure of the aircraft. Finally, LSTM does not directly include the spatial information for modelling the system's behaviour [62]. The proposed approach obtains the spatio-temporal dependencies using HOSVD for modelling the system behaviour. However, one limitation of the proposed approach is that it may not scale up well for high-dimensional and very nonlinear systems, especially those which require a large number of datasets (e.g., I >10,000).

In the next chapter, the co-optimization problem for design and control of system behavior will be provided.

Chapter 4: Data-Driven Prediction for Co-optimization: Applications to Design and Control of Traffic Road Networks

In this chapter, the outcome of the author's investigation due to Research Question 3 is presented, i.e., *how can the prediction of a system's behavior, using data collected from simulations or sensors, be used for co-optimization of a system's design and control?*

This chapter is organized based on the author's recent article[5]. Accordingly, the author begins with a nomenclature section specifically tailored for the material presented in this chapter. Then, in Section 4.1, a general statement of design and control co-optimization problem is presented, with a specific formulation of the traffic system problem. Next, the related literature review is presented in Section 4.2. An RL-based co-optimization approach is then proposed in Section 4.3. To demonstrate the proposed approach, in Section 4.4, the proposed co-optimization framework is applied to two road network examples, one for a three-legged intersection and another for four four-legged intersections. The results in both case studies are compared with traditional RL-based control only approach. Finally, a summary of the chapter is given in Section 4.5.

[5] Zhao, X., Azarm, S. and Balachandran, B. (2021) "Reinforcement Learning for Co-optimization of Vehicular Flow Direction Design and Signal Control Policy for a Road Network." *IEEE Transaction on Intelligent Transportation Systems.* [Under review].

Nomenclature for Chapter 4:

a_t	Vector of traffic signal control actions of size *m*, integer, at time *t*
$a^{'}$	Vector of successor actions of size *m*, integer, yielding highest Q-value
A	Action space
D	Stored data set
$e_{ii'}$	Edge indicates a vehicular flow direction from road v_i to road $v_{i'}$
E	Edge in directed graph
$E(\cdot)$	Expected value function
$G(\cdot)$	Directed graph representation
i	Index for intersection
I	Index for integer variables
k_i	Number of vehicular flow directions for the *i*th intersection
K	Total number of vehicular flow directions for a given road network
$L(\cdot)$	Loss function for *Q*-value update
m	Number of intersections in a road network
n_i	Number of roads for the *i*th intersection
N	Total number of roads for a given road network
$p(\cdot)$	Probability density function

$Q(\cdot)$	Q-value function
Q_π	Q-value function for control policy π
Q^*	Optimal Q-value function
$r(\cdot)$	Reward function
r_t	Reward at time t
R	Expected future reward
s_t	Vector of traffic states of size N, mixed continuous-integer, at time t
s'	Vector of successor state of size N, mixed continuous-integer
S	State space
t	Index for time step
T	Terminal time step
$U(\cdot)$	Uniformly drawn sample
v_i	ith vertex of graph representing ith road
V	Set of vertices in directed graph
x	Vector ($\mathbb{R}_I^{N+K}$) of integer design variables for vehicular flow directions
X	Vehicular flow direction design space
α	Learning rate
γ	Discount factor for learning: $\gamma \in [0,1)$

θ	Vector of parameters for neural network
θ^{-}	Vector of parameters for neural network at previous training iteration
π	Traffic signal control policy function
π^{*}	Optimal traffic signal control policy function

ACRONYMS

DQN	Deep Q-learning network
MDP	Markov decision process
RL	Reinforcement learning

4.1 Design and Control Problem Statement and Assumptions

In this section, a general problem formulation for prediction and co-optimization of system design and control is provided in Section 4.1.1. Then, an application to traffic road network co-optimization formulation is presented in Section 4.1.2. Meanwhile, associated assumptions for this problem are detailed.

4.1.1 Design and Control Co-optimization Problem via an Extension to Markov Decision Process

The aim of the problem is to predict and then maximize a measure of system performance, or reward, through co-optimization of the design and control of the system. This problem can be formulated using an extension of a Markov Decision Process (MDP) [104], as depicted in Figure 4.1. This MDP problem is defined by the tuple: $\{X, S, A, p, r, \gamma\}$, wherein the system observes the state of the environment $s_t \in$

S at time t and takes an action $a_t \in A$ according to a control policy $\pi(a_t|s_t, x)$. Here, the control policy π is a computable function that takes the state $s_t \in S$ and system design $x \in X$ as inputs and return the probability of taking an action a_t, where the action is chosen to be with maximum predicted of system performance (or reward). The design of a system governs the system controller's interaction with the environment through a transition probability function $p(s_{t+1}|s_t, a_t, x)$. Then, the system performance (or immediate reward) is dependent on the design as well: $r_t = r(s_t, a_t, s_{t+1}, x)$.

The objective is to maximize the expected future reward (system performance) R by co-optimizing the design x and control policy π as:

$$R^* = \max_{x,\pi} R\ (x, \pi), \tag{4.1}$$

where the expectation of future reward is given by:

$$R(x, \pi\ ) = E_\pi[\textstyle\sum_{t=0}^{T} \gamma^t\, r(s_t, a_t, s_{t+1}, x)|s_0 = s, a_t = \pi(s_t, x), x], \tag{4.2}$$

with an initial state s_0. The expected future reward is to be maximized with a prediction and learning-based approach.

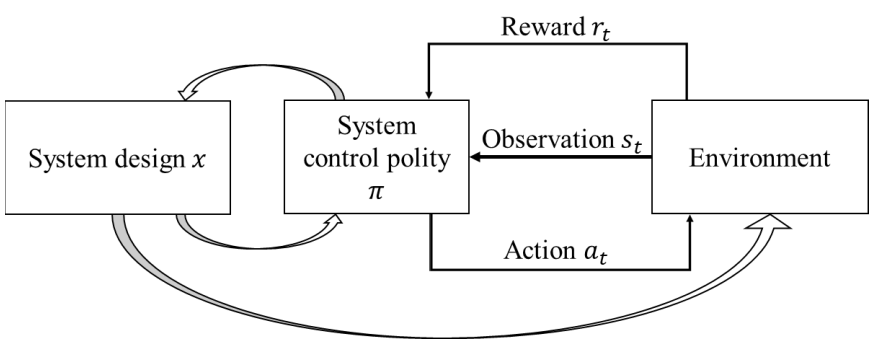


Figure 4.1 Co-optimization for design and control of a system: an extension of MDP.

4.1.2 An Application to Co-optimization of Traffic Road Network Problem

The aim in this application is to co-optimize the design of the vehicular flow directions along the roads and control the (red or green) traffic signals at one or multiple intersections while an overall traffic performance for a road network is predicted and maximized.

In the following, the definitions for traffic performance, vehicular flow direction design and traffic signal control, with corresponding assumptions, are provided:

Traffic performance/reward $(r \in R)$. The traffic performance is predicted by the weighted sum of two normalized criteria, wherein the weights are assumed to be predetermined (between 0 and 1) and set by a traffic authority (decision maker) according to his/her experience. The first criterion is a cumulative vehicle throughput for all intersections in a road network. The cumulative vehicle throughput is defined as the number of vehicles passing through all intersections per unit of time. The second criterion is vehicles' cumulative queue length per unit time for all intersections. The queue length is defined as the number of vehicles waiting per unit of time to be served by a traffic signal. The cumulative queue length is calculated as the sum of the worst (longest) queue length at each intersection, and summed up over all intersections at each time step.

Vehicular flow direction design (*x,* vector of size $N + K$). For a road, the vehicular flow direction can be designed to have a one-way or two-way direction. For a multi-legged intersection, a vehicular flow direction can be designed to determine a right of way (or restricted) direction of a vehicle from one road to another road. Examples of vehicular flow directions for a road are shown in Figure 4.2. Figure 4.2(a) shows a one-

way vehicular flow direction for the roads at a T-junction. Figure 4.2(b) shows two-way (opposite) vehicular flow directions for the roads at a four-legged intersection, and Figure 4.2(c) shows a mixture of one- and two-way vehicular flow directions for the roads at a five-legged intersection.

For a four-legged intersection (see Figure 4.2(b)), as an example, Figure 4.3 contains possible vehicular flow directions across the intersection. As shown in Figure 4.3(a) – 4.3(d), a vehicle at each of the four roads can have at most three right of way (or restricted) flow directions. For a right of way direction (sold black line), a vehicle has the option to make a left-turn, right-turn or go straight, or might have restriction (shown with dashed black line).

Consider the design of vehicular flow directions for the roads in a road network with m multi-legged intersections, wherein each intersection i has n_i roads with k_i vehicular flow directions across the intersection. Since there are a total of $N = \sum_{i=1}^{m} n_i$ roads in the network, and each road has three design options: two one-way (with opposite) vehicular flow directions and one two-way flow direction, then considering the roads

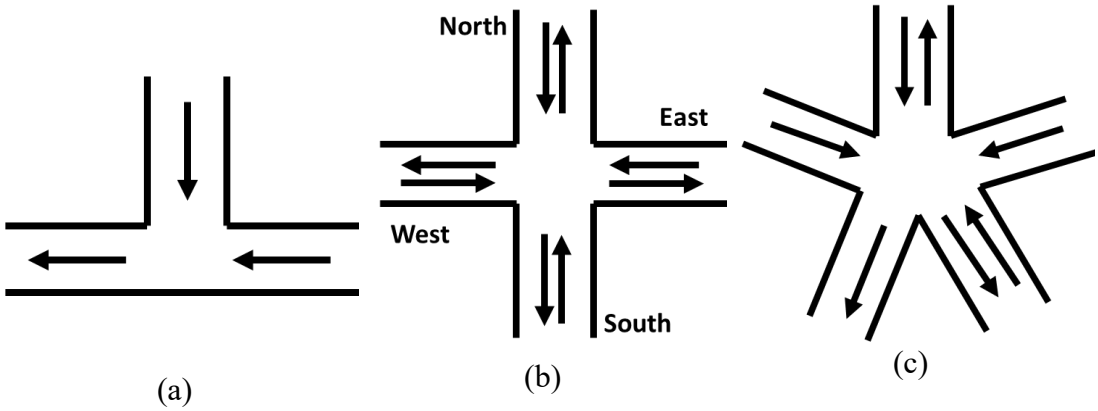
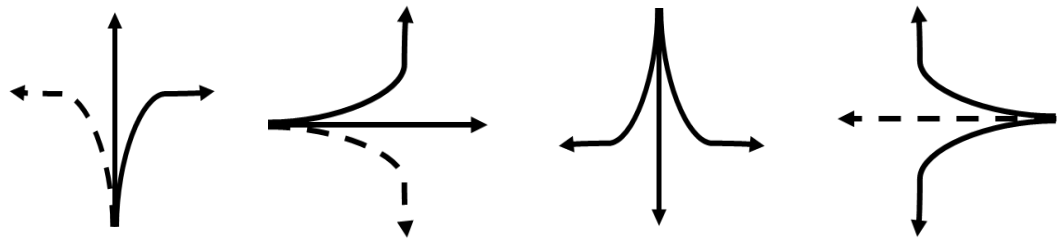


Figure 4.2 Examples of intersection types: (a) T-junction with three one-way roads, (b) Four-legged intersection with four two-way roads, an (c) Five-legged intersection with mixed one-way and two-way roads.



(a) (b) (c) (d)

Figure 4.3 Example of four traffic signal for the four-legged intersection (Figure 1(b)) composed of vehicular flows from (a) South, (b) West, (c) North, and (d) East roads. (A dashed arrow is used for a restricted (prohibited) direction, while a solid black arrow is for a non-restricted direction.)

the number of design options for vehicular flow directions is 3^N. For the design of vehicular flow direction across m intersections, there are a total of $K = \sum_{i=1}^{m} k_i$ vehicular flow directions, wherein each direction has two design options: a permitted direction or a restricted direction. Thus, for the design of vehicular flow directions across intersections, the number of design options is 2^K. Therefore, the maximum number of design options for the vehicular flow directions in a road network is $3^N \times 2^K$.

Traffic signal control (a_t, vector of size m). Given the vehicular flow direction design for a road network, a centralized traffic control policy π is assumed to control the traffic signal for all intersections. Each signalized intersection is pre-assigned with a set of phases. A phase can be defined as the right of way directions (green light) assigned to a combination of non-conflicting vehicular flow directions at an intersection. For example, in the four-legged intersection shown in Figure 4.2(b), there can be four signal phases as shown in Figure 4.3. In general, an n_i-legged intersection can be assigned with n_i number of signal phases. The goal of the traffic controller is to choose one signal phase at each time step. For a road network with m number of multi-

legged intersections, the traffic signal controller is used to decide on a combination of signal phases $a_t = (a_t^1, a_t^2, \dots, a_t^m)$ for each intersection. Thus, at each time *t*, the total number of signal phase combinations is $n_1 \times n_2 \times \dots \times n_m$.

The following assumptions are made: i) A subset of the roads in a road network is assumed to be "injection" roads. The injection roads are used by a simulator to inject vehicles into the roads as they approach an intersection. The injection rate for injection roads can be changed to simulate rush hour or non-rush hour scenarios. ii) A vehicular flow direction design is feasible if a vehicle that enters from any of the injection roads into the road network can exit from any of the roads. iii) If a road is one-way, it is assumed to have a single lane. If a road is bi-directional (two opposite ways), then each way has a single lane. iv) A centralized controller is used to determine a combination of traffic signal phases for all intersections as a function of time. It is assumed that a traffic signal phase duration (e.g., time for green light) is predetermined. vi) Simulation or sensors (e.g., inductive-loop traffic detectors located at intersection) are assumed to be able to collect the following traffic data, such as the number of vehicles passing through the intersection, distance of the lead vehicle from the intersection, and drivers' intentions (making a turn or going straight at an intersection) before arriving at the intersection. These data will be used to construct model for traffic performance prediction.

4.2 Related Literature Review

The ever-increasing growth in population and number of vehicles in major urban areas have resulted in significant traffic congestion and other related challenges for transportation authorities. Traffic congestion can increase vehicular queueing, travel

delays, fuel consumption, economic loss, and pollution [79]. Two key factors that contribute to traffic congestion in a road network are: (i) design of vehicular flow directions [80-82], and (ii) control policy used for traffic signals.

Decisions made for both the vehicular flow directions and traffic signal control policy can have significant effects on the traffic performance, and their effects are inherently coupled. Different vehicular flow directions can result in different traffic signal control policies and vice versa. One way to optimize the traffic performance for a road network, as presented in this chapter, is to consider simultaneously the design of vehicular flow directions and control policy for traffic signals.

In the literature, there are reports on two classes of techniques for handling this problem, namely, model-based optimization and model-free RL techniques. In reference [83], four model-based optimization approaches are proposed and studied to jointly find optimal road network topology and traffic signal controller. However, model-based techniques can rely on some strong assumptions and a specific traffic control model, and thus can be hard to generalize. On the other hand, model-free RL techniques [84-103] have two key benefits. The first benefit is that they do not have strong assumptions and a control policy is learned based on available data; for example, from GPS-equipped vehicles and loop detector sensors. For RL-based techniques, an optimal performance can be achieved through a prediction and interaction with the environment through a trial-and-error search scheme. The second benefit is that in high-dimensional state-action spaces, function approximation techniques in RL can be used to achieve computational efficiency [95].

In the literature, RL-based techniques have been reported to obtain some impressive results, but only for solving traffic signal control problems [84-103]. Indeed, the existing works on RL-based traffic signal control can be categorized according to different characterizations of state space, action space, reward function, RL algorithms, and corresponding applications. Traditionally, the state space has been defined in terms of real-time traffic information such as vehicular flow rate [84, 85], queue length [86, 87], and average delay time [88, 89]. There have been recent efforts to include image-like features [89-91] in the state to provide a more comprehensive description of the traffic conditions, with positions of vehicles along the lanes indicated by a binary matrix. In terms of action space, the action of the traffic controller is defined by determining the best traffic phase. Each phase refers to the right of way (green light) time duration assigned to a combination of vehicular flow directions. Generally speaking, the action space is of two types: i) step-based action [85, 92], in which the traffic controller, as an agent, is used to decide when to switch or stay in a traffic phase for a pre-determined time duration; and ii) phase-based action [84, 93-94], in which the agent is used to decide the time duration for each traffic phase. As to the reward function or traffic performance, it is typically defined as a weighted sum of several metrics, such as travel time [85, 90] and queue length [87, 88]. Meanwhile, different RL algorithms have been investigated for controlling the traffic signal, including Deep Q-learning Network (DQN) [88, 95-96], policy gradient method [97], and actor-critic method [98]. Performance of these algorithms has been evaluated based on various network scenarios, such as single intersection control [88], multi-intersection control [99-102], and even massive-scale control - the road network of Manhattan with 3,971

traffic signals [103]. Even though significant progress has been made with the use of model-free RL techniques for traffic signal control, all those techniques only focus on the optimization of the traffic signal controller assuming a pre-determined or fixed design of vehicular flow directions.

In this chapter, it will be demonstrated that the design of the vehicular flow directions and traffic signal control policies are strongly coupled and should be predicted and optimized together. For solving this combined problem, the traditional RL-based approaches, formulated as a standard Markov Decision Process (MDP) [104], do not allow for a way to find the "best" vehicle flow direction design during the learning procedure for the traffic signal controller. One way to resolve this problem is by decoupling the design and control problems and solving them by using a bi-level technique. In this way, first, a set of candidate vehicular flow directions can be generated for the road network. Then, the traffic signal controller is optimized for each design and in this way the best solution for the design and control is found. Unfortunately, such a technique requires a learning procedure for all candidate designs, which can become computationally intractable. It is the aim of this chapter to address this limitation and present a new RL-based approach to predict traffic performance and co-optimize the design of the vehicular traffic direction and control of traffic signal policy for a road network.

4.3 Proposed Reinforcement Learning Technique

In this section, first, a directed graph approach is presented which can be used to determine feasible vehicular flow directions in Subsection 4.3.1. Then, a co-optimization strategy in an RL setting is presented in Subsection 4.3.2.

4.3.1 Graph for Vehicular Flow Direction Design

The design of the vehicular flow directions for a road network can be represented by a directed graph $G(V, E)$, where V represents a set of vertices in which each vertex represents a road, and E represents a set of edges in which each edge $e_{ii'} \in E$ represents a vehicular flow direction from road v_i to road $v_{i'}$. In this way, a vehicular flow direction design for a given road network can be visualized as a directed graph with a set of vertices (roads) and edges (vehicular flow directions). For example, as plotted in Figure 4.4(a), a graph representation of a T-junction with three one-way roads (Figure 4.2(a)) is presented. In this graph, three vertices (solid black bullets) are for the three roads in the T-junction. The vehicular flow direction for these roads, as one-way roads, is represented by the edges. The dashed arrows (or edges) show restrictions for vehicular flow directions, while solid arrow edges show no restriction for vehicular flow directions. For example, Figure 4.4(a) is a graph representation of the T-junction (recall Figure 4.2(a)) in which there is no restriction on the vehicular flow direction from North-to-West or East-to-West. On the other hand, Figure 4.4(b), is a graph representation for a four-legged intersection with bi-directional roads (recall Figure 4.2(b)) with turn restrictions as shown in Figure 4.3.

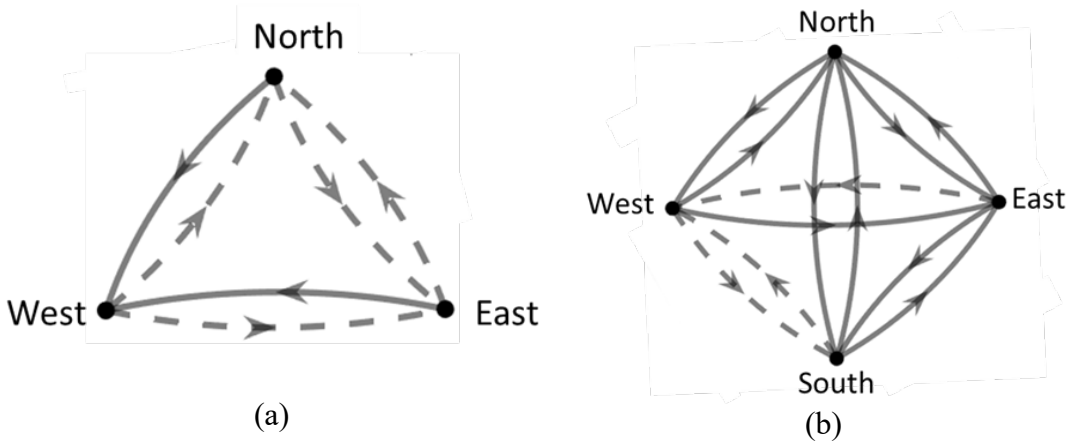


Figure 4.4 Graph representation of vehicular flow directions for (a) T-junction with three one-way road (Figure 1(a)), (b) Four-legged intersection with bi-directional roads (Figure 1(b)) and turn restriction (Figure 5.2). (A dashed arrow is used for a restricted (prohibited) direction, while a solid arrow is for a non-restricted direction.)

With the graph representation of vehicular flow directions, the feasibility of a design can be verified through the theory of the directed graphs [113], wherein the injection roads (vertices) are required to be "strongly connected" [109]. This is achieved by using the Kosaraju's algorithm [109] to ensure strong connectivity among the injection roads. In this way, a feasible set of vehicular flow directions can be determined.

4.3.2 Co-optimization of Vehicular Flow Direction Design and Traffic Signal Control

The DQN approach is extended to account for the design variables (vehicular flow directions) as additional inputs (in addition to the traffic signal controller inputs, or signal phases) in order to predict the Q-value function: $Q(s, a, x: \theta) \approx Q^*(s, a, x)$, wherein θ is for parameters of the neural network. As such, the objective function is formulated as:

$$\max_{x,\theta} E_{\pi}[Q(s, a, x: \theta) | a = \pi(s, x)] \tag{4.3}$$

Following this formulation, the problem for the vehicular flow direction design and the problem for traffic signal control are coupled. The two-way coupling in this formation is due to the following reason. The Q-value function can be used to predict the traffic control performance for different designs, and a prediction of the Q-value function can be used to eventually obtain an optimal vehicular flow direction design. Meanwhile, once a vehicular flow direction design is fixed, an optimal value of the Q-value function: $Q^* = \max_{\pi} Q_{\pi}$, can yield an optimal control policy $\pi^*(a_t|s_t, x)$, where the combination of signal phases for multiple intersections a_t is decided by obtaining the maximum predicted Q-value as $a_t \in \arg\max_{a_t} Q^*(s_t, a_t, x)$.

The co-optimization is done by using a co-learning approach, which trains the Q-value function in order to simultaneously optimize the vehicular flow direction design and traffic signal controller. Then, an iterative optimization process is followed to update the Q-value function. The iterative process is demonstrated in the flowchart of Figure 4.5, with a corresponding pseudo-code in Table 4.1, and described in the remainder of this subsection.

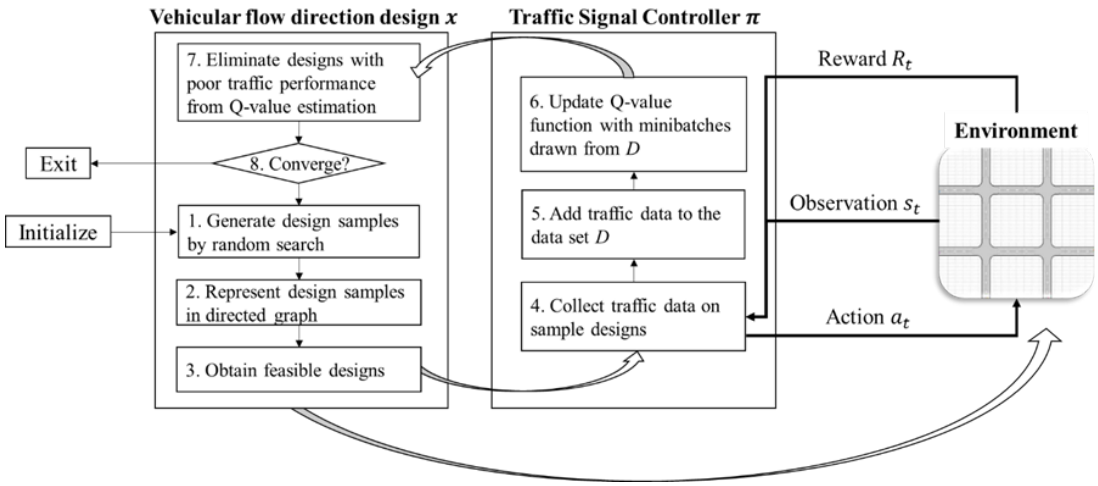


Figure 4.5 RL-based flowchart for co-optimization of vehicular flow direction design and traffic signal control.

Table 4.1 Pseudo-code for co-optimization of vehicular flow direction design and traffic signal control

Initialize data set: D
Initialize $Q(s, a, x: \theta)$
While not converge **do**
  1. Initialize first random vehicular flow design samples x
  2. Represent design sample in directed graph
  3. Obtain feasible designs
  for $i \in (1,\ 2,\ \ldots)$ **do**
    4. Collect traffic data $\{\ s_0, a_0, r_1, s_1 \ldots, r_T, s_T, x\}$ for a vehicular flow direction design x with policy π
    5. Add traffic data to date set D
    6. Update Q-value function with minibatches drawn from D
  end for
  7. Eliminate designs with poor traffic performance from Q-value estimation
  8. Check whether converge to one design option
End while

In the approach, initially, the data set D is set to be empty. The Q-value function is set to be approximated by using a neural network as $Q(s, a, x: \theta)$, wherein the parameters of the neural network θ are randomly initialized. After the start of co-learning procedure, first, the vehicular flow direction designs are randomly sampled (Step 1 of Figure 4.5 and Table 4.1). This is done by a discrete representation of the

design variable (flow direction designs). For example, vehicular flow direction design for a road is indicated by 0 (clockwise one-way), 1 (two-way), and 2 (anticlockwise one-way). Similarly, vehicular direction design across intersections is indicated by either 0 (prohibition of the move), and 1 (allowance of the move). Random integer generator is used to produce random design alternatives. Next, those design samples are represented by a directed graph (Step 2), as detailed in Section 4.3.1. With the search algorithm mentioned in Section 4.3.1, a set of feasible design alternatives are obtained (Step 3).

Next, a history of the traffic data will be collected for the feasible vehicular flow directions through the interaction with the road network environment (e.g., using sensors) (Step 4). The data collection process is as follows: for a training iteration, the data is collected for one vehicular flow direction design from an initial time step of 0 to a terminal time step *T*. At each time step *t*, a vector of traffic states s_t is collected, including the number of vehicles on each road, distance of the lead vehicle from the intersection, and driver's intention from each road. Then, a centralized traffic signal controller is used to decide on a combination of signal phases a_t for all intersections according to the current control policy $\pi(a_t|s_t, x)$ such that $a_t \in \arg\max_{a_t} Q(s_t, a_t, x)$. In the meantime, an immediate traffic performance or reward r_t, and the traffic state for the next time s_{t+1} are observed. In this way, a data point composed of $(s_t, a_t, r_t\ , s_{t+1})$, is obtained at each time step. After completing one training iteration, a set of traffic data for a vehicular flow direction design is collected as $\{\ s_0, a_0, r_1, s_1 \dots s_{T-1}, a_{T-1}, r_T, s_T, x\}$ (Step 5). In this way, multiple sets of traffic data

are obtained after multiple training iterations. This set of data is stored in a data set D (Step 6).

In the next step, the Q-value function is updated with the collected traffic data through the experience replay technique [110] (Step 6). A minibatch of the traffic data is uniformly drawn at random from the stored traffic data D. Then, the stochastic gradient-decent method [110] is used to update the Q-value function parameters θ by minimizing the following loss function:

$$L(\theta) = E_{(s,a,r,s')\sim U(D)}\left[\left(r + \gamma \max_{a'} Q(s', a', x: \theta^{-}) - Q(s, a, x: \theta)\right)^2\right] \tag{4.4}$$

Here, θ^{-} are the parameters of the neural network from the previous training iteration. During the procedure, the Q-value function is updated based on data from a diversified set of vehicular flow direction designs and corresponding traffic signal control.

Next, in Step 7 the updated Q-value function is used to approximate the traffic performance on a sample of feasible designs obtained from Step 3. The estimation of traffic performance for a single design x is calculated by the summation of the Q-values for a set of traffic states given the current control policy, as shown in equation (4.5):

$$\sum_{s} Q(s, a, x: \theta | a = \pi(s, x)) \tag{4.5}$$

In the above equation, s is a set of traffic states which is randomly generated. After obtaining an estimation of the traffic performance for a sample of vehicular flow direction designs, a portion (or percentage) of the design alternatives with the lowest estimated traffic performance is removed from further consideration for the following training steps. The choice of the eliminated portion (e.g., 50%) is predetermined by a user to balance the speed of the convergence and quality of the solution obtained. For

example, a large percentage can lead to convergence to a design with fewer training iterations while at the same but compromise the quality of the design solution obtained.

The final step is to check whether there is only a single vehicular flow direction design remaining (Step 8). If not, the approach is used to explore the remaining design candidates, and then sample another round of design alternatives through a random search strategy. This process is repeated (back to Step 1) until after several iterations, only a single vehicular flow direction design is obtained. Once there is only a single design left, the traffic signal controller is trained (Step 4 to 6) using that remaining design until there no change in the traffic performance is observed.

4.4 Examples

The performance of the proposed approach is evaluated by using a traffic simulation tool OpenTrafficLab [115], which is built in MATLAB^R [116] using the Automated Driving Toolbox^{TM}[117]. Two examples with different types of road networks are considered. In Example 1, a three-legged intersection (T-junction) is used to demonstrate how the proposed approach works step-by-step. In Example 2, a road network composed of four, four-legged intersections is considered. Example 2 is used to demonstrate an application of the proposed approach with one- and two-way (bi-directional) vehicular flow directions and traffic signal controller for multiple intersections. By using these two examples, the traffic performance of the proposed co-optimization approach is compared with that obtained for the RL-based traffic signal controller without the consideration of the vehicular flow direction design. In these examples, the traffic performance is formulated as the weighted sum of the vehicular queue length and throughput, where in the weights are arbitrary chosen as 0.1 and 0.9,

respectively. The weights are selected to equalize the effects since quantitatively the vehicle queue length can be significantly higher than the throughput.

4.4.1 Three-legged Intersection

The geometry for a three-legged intersection is illustrated in Figure 4.6, in which the three roads are connected through an intersection. For this example, the simulated vehicles are injected from the open end of each road (shown with arrows in Figure 4.6 from West, East, and South sides, as injection roads), with a Poisson distribution which is used to specify the number of vehicle arrivals at each road in a certain time period. A vehicle flow rate of 600 or 200, respectively for rush hour or non-rush hour, is chosen to simulate the expected number of vehicles per 3,600 time steps coming from the injection roads. The drivers' intentions are randomly sampled, with equal probabilities for non-restricted vehicular flow directions across the intersection, e.g., going straight, turning left, or turning right.

In this example, as shown in Figure 4.6, each road has a two-way vehicular flow direction, which is fixed (not considered as a design variable). The design variables for this example are defined as the six vehicular flow directions through the intersection, as shown in Figure 4.7, which are composed of two vehicular flow directions from each of the West, South, and East sides. Each design variable has two integer values (0 or 1), with 1 being used for permitting (right of way of) a vehicular flow direction through the intersection (from one road to the other), and 0 for prohibiting it. Thus, the maximum number of vehicular flow direction design alternatives is: $2^6 = 64$.

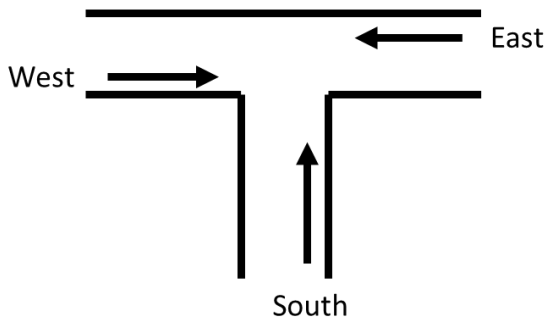


Figure 4.6 Geometry of three-legged intersection.

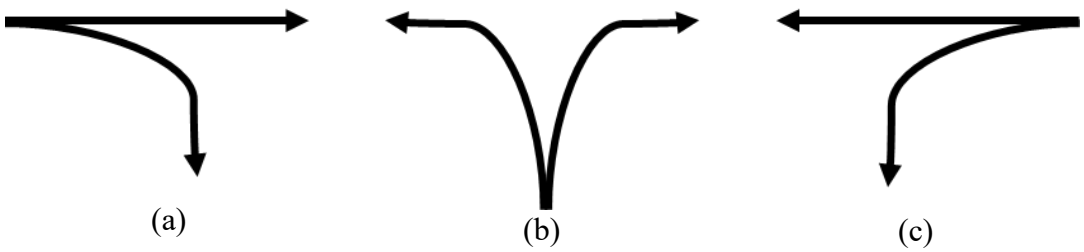


Figure 4.7 Three traffic signal phases for the three-legged intersection, composed of vehicular flow directions from (a) West (phase a), (b) South (phase b), and (c) East (phase c).

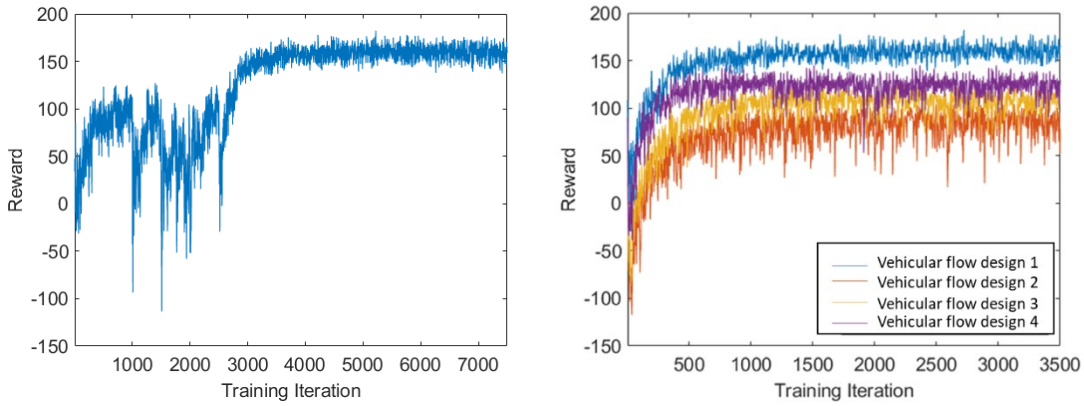
On the other hand, the control variable is defined as the decision on the traffic signal phase for the intersection. Three traffic signal phases are defined, as in Figure 4.7(a)-(c), which correspond to the green light assigned to vehicular flow directions from the West, South, and East end of the roads, respectively. The time duration for each signal phase is predetermined to be 40-time steps. The objective of the traffic signal controller is to choose one of the three traffic signal phases every 40-time steps from an initial time step of 0 to a terminal time step of 2000. Thus, the total number of control options in this example is equal to: 3^{50}.

Following the application of the proposed technique, initially, there were 64 design options for the vehicular flow directions (Step 1). Then, the directed graph approach (recall subsection 4.3.1) was used to represent these design alternatives and verify their feasibilities (Step 2). By using the graph algorithm, it was determined that 19 of the

alternatives had feasible directions. Next, 500 design samples were randomly generated out of the 18 feasible design options (Step 3). Each vehicular flow direction design was used to collect a set of traffic data. For every training iteration, the traffic signal controller's interactions with the three-legged intersection environment (using the simulator) were used to collect a history of the traffic data for each of the vehicular flow direction design alternatives under consideration (Step 4). These traffic data were then stored (Step 5). At the end of each training iteration, the parameters of the Q-value function were updated with the collected traffic data by using equation 4.10 (Step 6). After repeating the procedure (Step 4 to Step 6) for 500 training iterations, traffic signal control policy associated with the Q-value function was improved accordingly. The traffic performance for the training procedure is visualized in Figure 8(a). The updated Q-value function was used to estimate the traffic performance for the 19 feasible designs, from which 9 designs with the lowest traffic performance were eliminated in the following training iterations (Step 7). The training procedure was then continued (Step 8, and back to Step 1) to explore the 10 remaining design options and the iterations continued until only one design is left. The final design for the vehicular flow direction is shown in Figure 4.10 as "design 1". The process during downsizing to a single design (design 1) is shown in Figure 4.8(a) for the first 3,000 iterations, wherein there are significant fluctuations in the traffic performance (or reward) values. These fluctuations occur because of the exploration of different vehicular flow direction designs and learning process needed to find a traffic signal controller that could accommodate those designs.

Starting after the 3,000 iterations, only one design for the vehicular flow direction remained, and the traffic signal controller was optimized for that design. This is the reason why a smoother increase of the reward over the last 4,000 iterations is observed in Figure 4.8(a). Finally, the procedure converged to a solution, which resulted in a flat curve in the reward for the last few hundreds of training iterations. An example of the optimized traffic signal control solution is shown in Figure 4.9. A sequence of traffic signal phases (or actions) is determined for an instance of the traffic state and vehicular flow direction design 1 (shown in Figure 4.10). For example, in the first 40-time steps, phase (a) (shown in Figure 4.7(a)) is chosen when the West road expects a high traffic volume, while other roads have a low traffic volume. Here, the phase (a) is indicated by green band (for green light) for the West road, and red band (for red light) for the South and East roads, as shown in Figure 4.9.

Finally, the solution from the co-optimization procedure is compared against the RL-based traffic signal control-only approach. Three feasible vehicular flow directions are randomly chosen for this comparison, for which their configurations are plotted in Figure 4.10 as design alternatives 2 to 4. The optimization of the traffic signal control-only performance for each vehicular flow direction design alternative is illustrated in Figure 4.8(b), Each curve in Figure 4.8(b) corresponds to a case when a design is fixed, and traffic signal controller is optimized only for that design. As shown in Figure 4.8(b), the optimization of traffic signal control performance for different designs shares a similar trend; however, design 1 is found to have a better traffic performance than the other three designs.



(a) (b)

Figure 4.8 The offline training performance for Example 1 for (a) co-optimization (b) control only optimization under different vehicular flow direction designs.

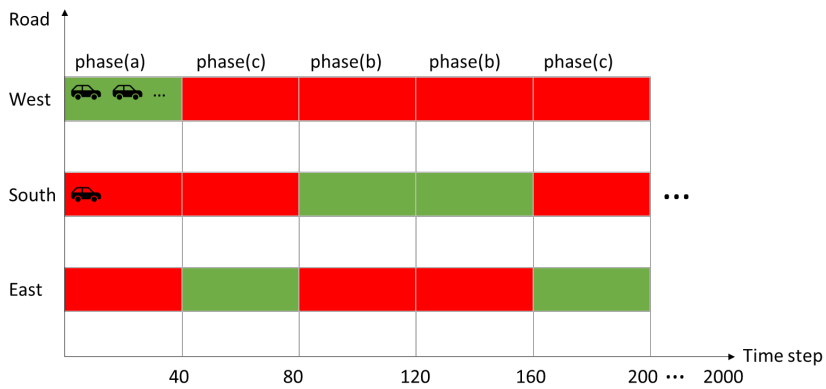


Figure 4.9 A sequence of traffic signal phases (or actions) for Design 1 (shown in Figure 4.10), and an instance of traffic state. For example, phase (a) is chosen for the first 40-time steps when the injection road from the West road expects to have a high traffic volume, while other roads have a low traffic volume. Green band is for the green light and corresponds to right of way for one road, while the red band is for red light and corresponds for stopping phase for other roads.

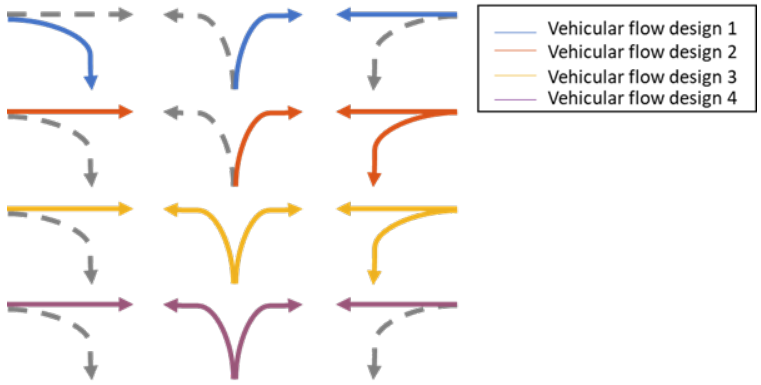


Figure 4.10 Vehicular flow directions through the intersection from West (left, phase a), South (middle, phase b)), and East (right, phase c) road for different design options. (A dashed arrow is used for a restricted (prohibited) direction, while a solid arrow is for a non-restricted or right of way direction.)

4.4.2 Four four-legged intersections

In this example, a road network is considered which comprise of four connected four-legged intersections consisting of twelve roads, as shown in Figure 4.11. Amongst these twelve roads, eight are injection roads, which are marked by arrows and numbered 1 to 8. Each injection road is bi-directional: has one incoming lane and one outgoing lane. Like in Example 1, two vehicle flow rates of 600 (rush hour) and 200 (for non-rush hour) are used for the expected number of vehicles per 3,600 time steps. In this example, each of the four intersections in the road network has 12 vehicular flow directions through the intersection, which are illustrated in Figure 4.12(a) – (d). These vehicular flow directions are assumed to be fixed (not considered as a design variable). Instead, the design variables are for deciding whether to have one, or two-way vehicular flow directions for the four connecting roads, indicated by East, South, West, and North roads in Figure 4.11. For each of the four roads, the vehicular traffic flow direction is selected to be 0 for clockwise one-way direction, 1 for two-way directions, and 2 for counterclockwise one-way direction. Thus, considering the vehicular flow direction

design for the four roads, the maximum number of design alternatives is $3^4 = 81$ for this example.

For this example, the control variables are decisions on a combination of traffic signal phases for the four intersections. For each intersection, four traffic signal phases are defined, as plotted in Figure 4.12. Each traffic signal phase corresponds to a set of non-conflicting vehicular flow directions that are assigned with green light. Similarly, as in Example 1, the time duration for each signal phase is predetermined for 40-time steps. In other words, the traffic signal controller is to choose a combination of traffic signal phases for four intersections (one phase for each intersection) for every 40-time steps, from an initial time step 0 to a terminal time step 2000. Thus, at every 40-time steps, the total number of control options (combinations of signal phases) for all four intersections is $4^4 = 256$. Therefore, for a sequence of signal phase combinations (every 40-time steps from time step 0 to time step 2000, which is 50 instances of 40-time steps), the total number of control options is 256^{50}.

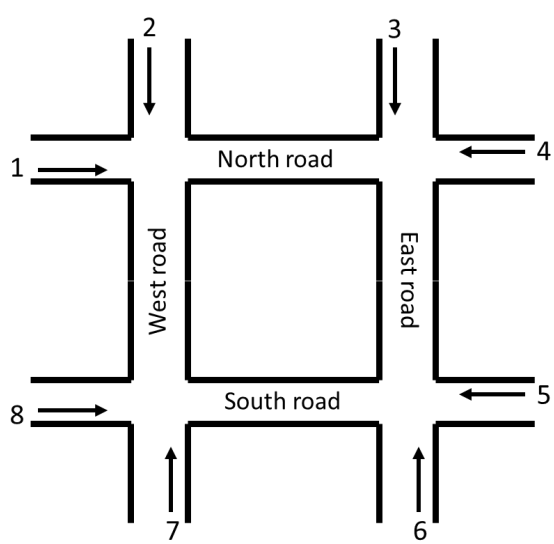
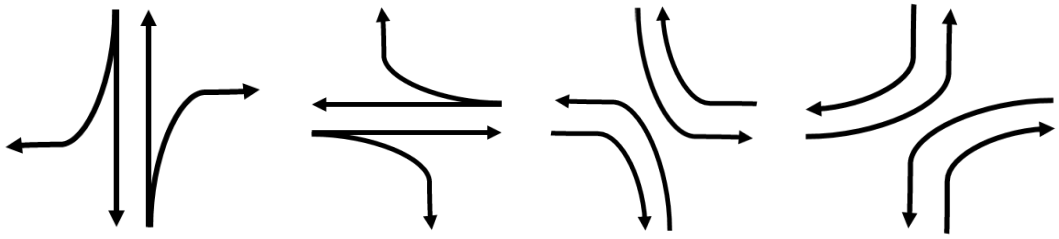


Figure 4.11 The geometry of four connected four-legged intersections.

Two scenarios are considered for this example: in the first scenario, injection roads 1 and 2 (Figure 4.11) is set to have a high-volume (rush hour) traffic, while the remaining injection roads have a low-volume (non-rush hour) traffic. In the second scenario, all eight injection roads are chosen to have a high-volume traffic. The goal in this example is to demonstrate how the vehicle flow direction design and traffic signal control adapt to different traffic volumes and how the traffic performance compares with that of the traffic signal control only.

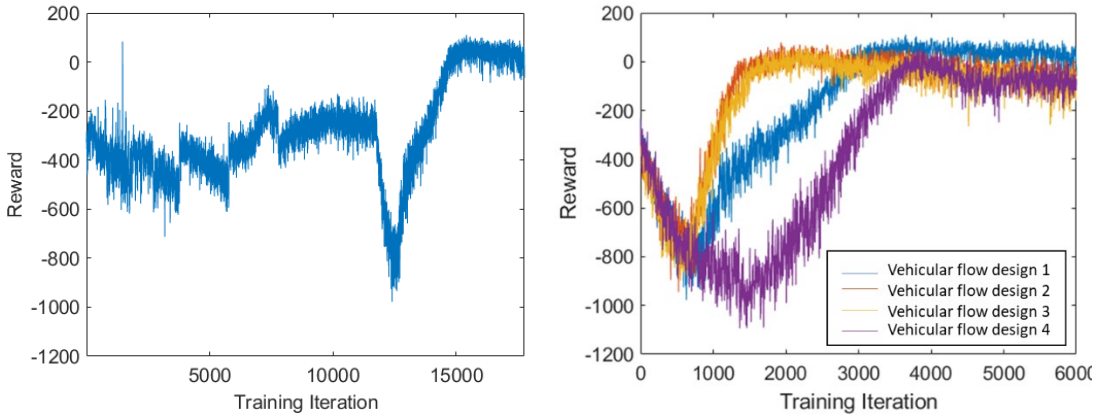
Following the proposed approach, the training procedure for the first scenario is demonstrated in Figure 4.13(a). Initially, there are 81 available design options. After verifying the feasibility through the directed graph approach (recall subsection 4.3.1), 31 design alternatives are determined to be feasible. For the first 12,000 training iterations, random design samples are generated out of 31 feasible design options, and traffic signal controller tries to learn to improve for a diversified set of designs, which is the main reason for the reward fluctuations shown in Figure 4.13(a). Starting from the 12,000 training iterations, the vehicular flow direction design is fixed (design 1configuration shown in Figure 4.14), and traffic signal control is optimized for that design. The improvement of traffic performance (reward) for the traffic signal controller is demonstrated for the last 6,000 iterations in Figure 4.13(a). Finally, the convergence is observed for a stable traffic signal control solution. An example of the optimized traffic signal control solution is visualized in Figure 4.15, wherein a sequence of signal phase combinations is decided based on an instance of traffic state. For example, in the first 40-time steps, a combination of phases, a, b, d, and c (as in

Figure 4.12) is decided for the four intersections from upper left, upper right, lower left, and lower right intersections of Figure 4.11, respectively.



(a) (b) (c) (d)

Figure 4.12 Four traffic signal phases for each of the four-legged intersection: (a) straight and right turn from North and South (phase a), (b) straight and right turn from West and East (phase b), (c) left turn from North and South, and right turn from West and East (phase c), (d) left turn from West and East, and right turn from North and South (phase d)



(a) (b)

Figure 4.13 The offline training performance for Example 2 for (a) co-optimization. (b) control only optimization under different vehicular flow direction designs.

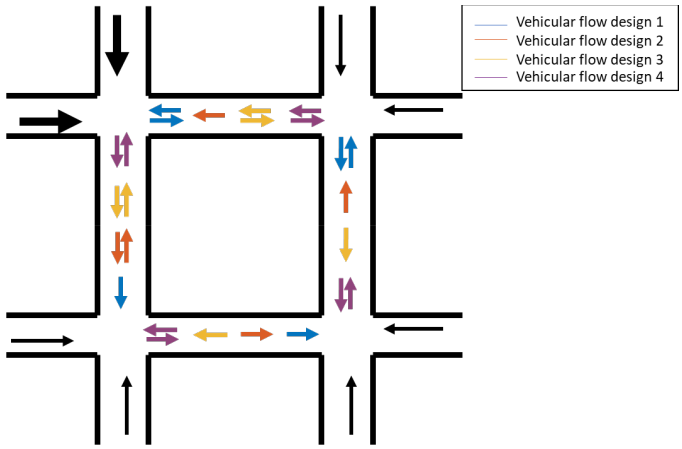


Figure 4.14 The vehicular flow direction design configurations for the roads for designs 1 to 4.

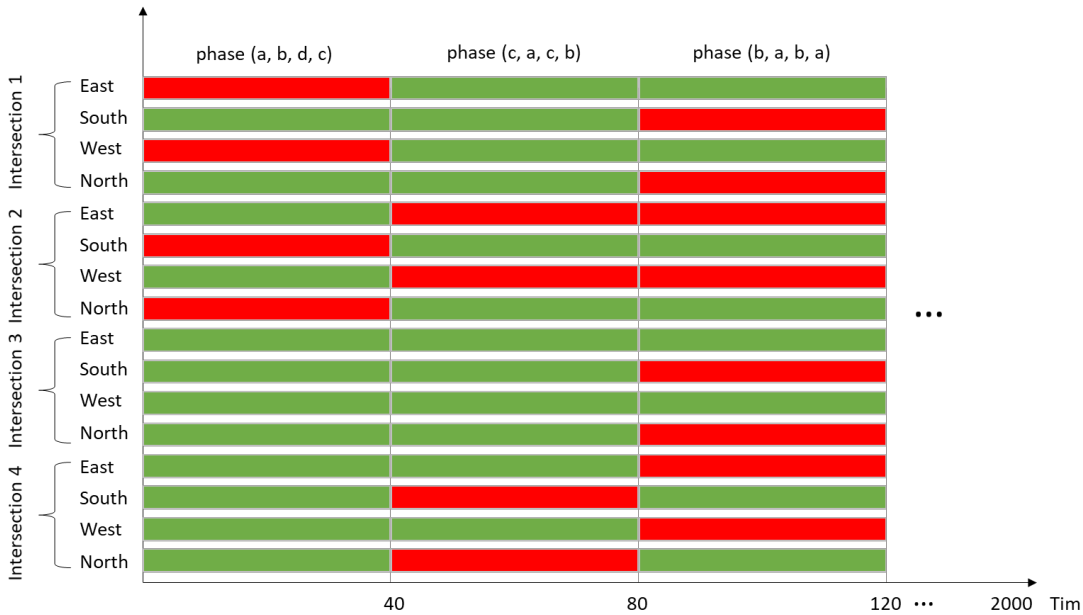


Figure 4.15 An example of a traffic signal control solution in the road network with four intersections, where a sequence of signal phase combinations is determined based on an instance of traffic state. Here, a signal phase combination is composed of four signal phases corresponding to intersections from upper left (1), upper right (2), lower left (3), and lower right (4) as shown in Figure 4.11. Each signal phase refers to right of way phase (green band) assigned to a mixture of roads (as shown in Figure 4.12.)

In Figure 4.13(b), the optimization of the traffic signal control performances for different vehicular flow direction designs are plotted, wherein the design is predetermined for each curve. In Figure 4.13(b), design 1 is obtained by the co-

optimization procedure, Figure 4.13(a), while designs 2 to 4 are random samples and their configurations are shown in Figure 4.14. The traffic signal control performance for different designs shares a similar trend, but design 1 shows a better reward or traffic performance than the other design alternatives.

For Example 2, online traffic experiments are simulated to evaluate the traffic performance for the four designs and their associated traffic signal controller. The traffic performance is evaluated for two objective functions: (i) sum of the vehicle throughput for all intersection (i.e., the total number of vehicles passing through all intersections), and a cumulative queue length which is calculated as the longest queue length at each intersection, and then summed up over all intersections at each time step. As plotted in Figure 4.16, cumulated vehicle queue length (Figure 4.16(a)) and throughput (Figure 4.16(b)) for each design is plotted for 2,000-time steps. The vehicular flow direction design 1 (obtained from co-optimization) is found to have the best performance, which yields the minimum queue length and maximum throughput after 2,000 time steps. The reason that design 1 is a better option in the first traffic scenario is as follows: when a high traffic volume is injected into road 1 and road 2 (for road numbers, see Figure 4.11), West road is designed as one-way from north to south in order to reduce the traffic flow to the intersection at the North-West intersection. Meanwhile, if the North road is designed as a one-way road from west to east to further reduce the traffic flow to the North-West intersection, that would result in an infeasible design option, as vehicles cannot exit from arrowed roads of 1 and 2 of Figure 4.11.

For Example 2, the second traffic scenario is the road network that is experiencing high traffic volumes coming from all injection roads. The co-optimization training

procedure for the second scenario is shown in Figure 4.17(a). For this example, the first 13,000 training iterations is the stage for exploring different design options and getting rid of designs with poor traffic performance behaviors. The last 8,000 training iterations are for the stage in which only one vehicular flow direction design remains, with its configuration plotted in Figure 4.17 as design 1 (different from design 1 of scenario 1, see Figure 4.18). For design 1, the four connecting roads are all having clockwise one-way direction. Additionally, the traffic signal control performance is compared among the vehicular flow direction design 1 and the three other randomly selected design alternatives as plotted in Figure 4.17(b). The three vehicular flow direction design configurations are illustrated in Figure 4.18 as design 2 to 4.

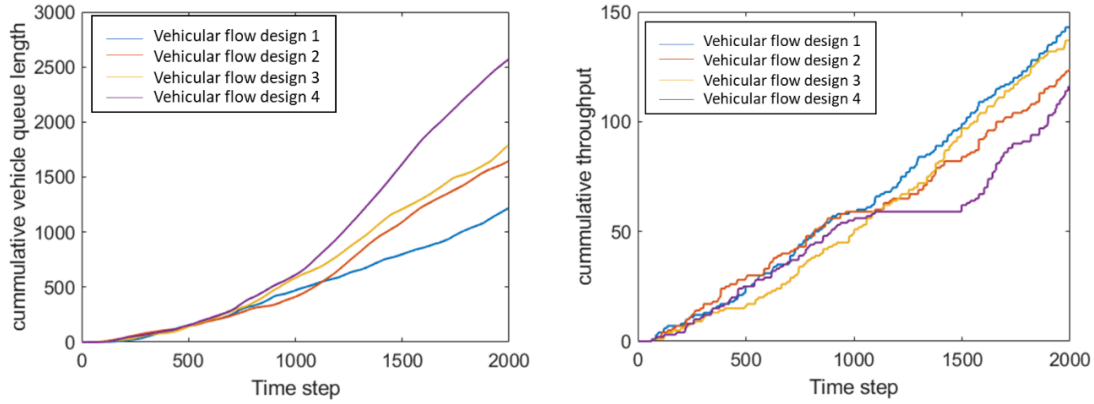


Figure 4.16 Online (after training) (a) cumulative vehicle queue length, and (b) throughput for different vehicular flow designs in first scenarios.

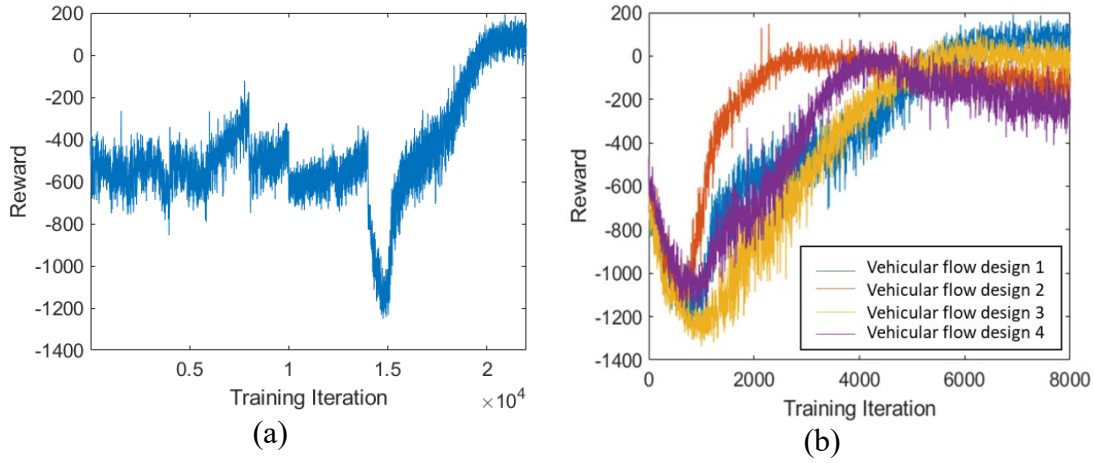


Figure 4.17 The offline training performance for (a) co-optimization process; (b) control only optimization process under different vehicular flow designs.

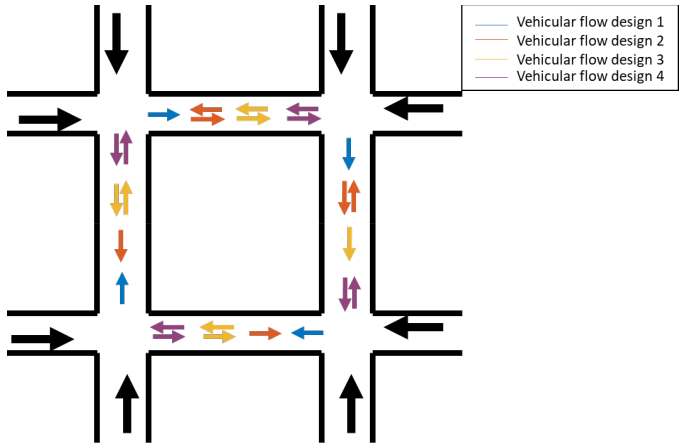


Figure 4.18 The vehicular flow configurations for different design options.

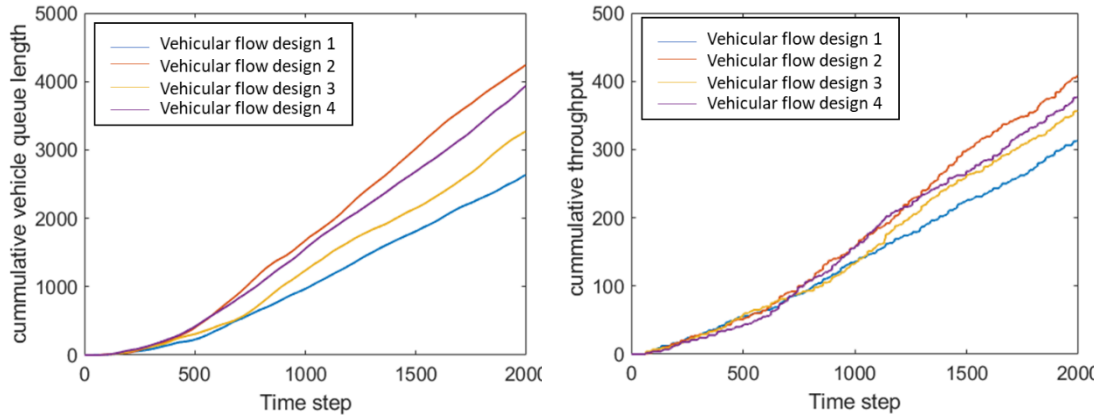


Figure 4.19 Online (after training) (a) cumulative vehicle queue length, and (b) throughput for different vehicular flow designs in second scenarios.

A video demonstration of the offline and online procedure for scenario 2 of Example 2 can be found in the following YouTube Link:

https://www.youtube.com/watch?v=hLjlqII-_E0.

4.5 Summary

The overall aim of this chapter has been to predict and improve the traffic performance for a road network, through co-optimization of the design of vehicular flow directions and control of traffic signal. To that end, the main contribution is the construction of an original RL-based approach, in which, the design of the vehicular flow direction problem is integrated into the RL-based traffic signal control problem. Through this approach, first, a directed graph is used to model the traffic direction design alternatives and check the design feasibility. Then, the DQN is extended to explore different design options with a random search algorithm. Simultaneously, the approach allows one to optimize traffic signal control and eliminate design options that have underperformed with respect to the traffic performance. Eventually, through learning, the approach is found to converge to a best combination of vehicular flow direction design and traffic signal control policy. The proposed approach is demonstrated through two examples: i) in Example 1, a step-by-step application of the proposed co-optimization procedure is demonstrated and used for the vehicular flow direction design and signal control for a three-legged intersection, and ii) in Example 2, the focus is on the design of the vehicular flow directions along multiple roads and traffic signal control of four, four-legged intersections. With the obtained results, the algorithm is demonstrated to have the ability to improve traffic performance with respect to throughput and queue length. Furthermore, it is shown that the proposed

approach outperforms the traditional RL-based traffic signal control-only approach. There are several advantages in using the proposed approach. This is purely a data-driven approach. Comparing against model-based approaches, the proposed approach has relaxed the assumption about underlying vehicle routing model or traffic dynamic model. This makes it easier for generalization. However, one limitation of the proposed approach is that a centralized agent is used which may not scale up well with a very large road network. That is because the number of design alternatives and control options can exponentially grow with number of roads and intersections in the given road network.

In the next chapter, the conclusion chapter, a summary of the thesis highlights, contributions, and possible future directions are presented.

Chapter 5: Conclusion

In this chapter, the author presents remarks to close the dissertation. In Section 5.1, a summary of the models and methods developed in this dissertation is provided, and some key results are highlighted. In Section 5.2, the main contributions of this dissertation are provided. Finally, some limitations of the current work and suggestions for extending the dissertation research are presented in Section 5.3.

5.1 Summary

The summary and highlights of the results for each research question are presented in this section.

First, a multi-step-ahead prediction problem is defined and formulated. For this problem, it is assumed the prediction is based on data collected from a single-response, high fidelity system simulation which is then fused with data collected from single-sensor measurements. Following the formulation of the problem, a dynamical data-driven prediction approach is proposed to estimate the system's behavior multiple time steps ahead. This new approach is called RECursive Multi-Input-Multi-Output (REC-MIMO), which can attain real-time estimation through a combination of expensive simulation data (offline) and sensor measurements (online). In the offline phase, a MIMO strategy is employed for constructing a predictive model for the system's dynamical responses from the simulation data. The offline phase is accomplished through a combination of proper orthogonal decomposition and Gaussian process. By utilizing the measurement data collected from single-sensor measurements, a recursive strategy is then used to iteratively enhance multi-step-ahead predictions by using a

particle filter method during the online phase. An illustrative example involving aeroelastic responses of a joined-wing SensorCraft is used to demonstrate the effectiveness of the proposed framework. It has been shown that the system's state predictions obtained through the REC-MIMO strategy have comparable accuracy to those obtained from high-fidelity simulations with significant reduction in the computational effort.

Second, the problem for prediction of multi-response system behavior with simulation data and sparse, noisy multi-sensor measurements (or physical experiments) is presented. The prediction in this case can be used for evaluating different system designs and operations. The prediction approach is constructed using a combined offline-online methodology, and is based on an integration of dimension reduction, surrogate modeling, and data assimilation techniques. A step-by-step application of the proposed approach is demonstrated through a numerical example. The performance of the approach is also evaluated by using two engineering examples. One of these examples involves the prediction of aeroelastic responses from a joined-wing aircraft in which sensors are sparsely placed on the aircraft's wing. The other engineering example involves predicting robotic appendage interaction with granular material and evaluating different appendage designs and operations. Through these two engineering examples, it is shown that the results obtained from the proposed prediction technique have comparable accuracy to those obtained from high-fidelity simulations, while at the same time significant reduction in computational cost is achieved. It is also shown that, for the case studies, the proposed approach has a prediction accuracy that is relatively robust to the limited sensor (experiment) data.

Third and finally, the problem of prediction and co-optimization in the design and control of the system's behavior is formulated and presented, along with the corresponding solution approach. The approach in this case is demonstrated with an application in predicting and optimizing traffic performance for road networks. To do that, a new RL-based technique is presented for co-optimization of the design of vehicular flow directions and control policy for traffic signals. This technique consists of a two-step iterative process. In the first step, a set of vehicular flow directions for a road network is generated. In the second step, an RL-based technique is used to train the traffic signal control policy over the given set of vehicular flow directions and predict traffic performance for each of the design options. Following the proposed technique, the vehicular flow directions with inferior predicted traffic performance are iteratively eliminated, while new vehicular flow directions are generated to achieve better traffic performance and realize convergence to a maximum possible expected traffic performance. The performance of the proposed RL-based technique is evaluated by using two examples under traffic conditions of rush hour and non-rush hour. It is shown that with the proposed RL-based prediction and co-optimization approach, a better traffic performance can be obtained, in terms of vehicular queue length and throughput compared with a conventional control-only approach.

5.2 Contributions

As a result of the research conducted for this dissertation, the author has made multiple contributions, including those presented in the following:

- The first main contribution made is the following: For the first time in the literature, a novel prediction approach is presented for forecasting a system's

dynamical behavior multiple timesteps into the future through an integration of data obtained from a high-fidelity system simulation and noisy sensor measurements. The proposed approach has the following key characteristics: i) one can achieve computational efficiency, close to real time, while achieving a comparable accuracy to that of an associated high-fidelity simulation; and ii) the prediction quality can be quite robust to different sensor noise levels.

- The second main contribution is on predicting multi-response system behavior with limited, and noisy multi-sensor measurements (or physical experiments). The key element for this contribution is the construction of a novel data-driven framework, which addresses three major challenges: i) predictive modeling of complicated spatio-temporal dependencies; ii) accounting for sparse, noisy sensors and experimental measurement data; and iii) significantly reduction of the computational expense. One can use the proposed prediction approach to evaluate multi-response system behavior and achieve an accuracy that is comparable to a high-fidelity simulation, and as shown in the case of an example, outperform the simulation and obtain results that are closer to the physical experimental data reported in the literature. It is envisioned that the proposed prediction approach can be used to effectively support offline system design and online operation even in the presence of a limited number of sensors or experiments data.
- The third main contribution is in the construction of a new problem and solution technique for prediction and co-optimization of the system design and control in the context of traffic systems. The main novelty here is that, for the first time in

the literature, a new reinforcement learning-based approach is constructed, with co-optimization of the design of vehicular flow directions and control of traffic signals. The results from this research are impactful because of the following: i) the proposed (model-free) approach is much easier to generalize, when compared to conventional model-based techniques. The proposed model-free approach does not require any strong assumption commonly considered in the model-based techniques (e.g., vehicle dynamics); ii) the proposed approach can be used to further improve traffic performance compared to the existing reinforcement learning-based approaches. This is achieved, for the first time in the literature, by simultaneously accounting for the effects of vehicular flow direction design and traffic signal control in the context of reinforcement learning setting.

5.3 Limitations and Future Directions

In this dissertation, the approaches presented in Chapters 2 and 3 can be applied to predict nonlinear systems' behavior while handling very limited sensor/experimental data with uncertainty. However, one limitation of the proposed approach is that it may not scale up well for very high-dimensional and nonlinear systems, which will require many datasets and highly expensive computational effort. On the other hand, senor or experimental data can be discrete and noisy. To address this shortcoming, a future study can be considered along this direction to develop an active learning approach for collecting simulation data, and then fused with sensor/experimental data, where a learning algorithm can interactively query the high-fidelity simulator for data in specific design and operation scenarios. In this way, it might be possible to reduce the

amount of data needed, while at the same time improve predictive capability of the approach.

Also, the proposed reinforcement learning-based approach, which was used for the traffic system problem can be generalized and applied for offline design and online control of a system. However, an important limitation of the proposed approach is that a centralized scheme was used for the proposed approach, which is difficult to scale up. For example, as shown in the traffic problem, the number of design alternatives and control options can exponentially grow up with the number of roads and intersections for a given road network. One possible future direction is to implement the proposed co-optimization approach in a quantum computing setting to extend its scalability.

Bibliography

[1] Nangia, R. K., Palmer, M. E., and Tilmann, C. P. (2003). “Unconventional high aspect ratio joined-wing aircraft with aft and forward swept wing tips.” *Proceedings of the 41st Aerospace sciences meeting*, pp.605, Jan. 6 -9, Reno, Nevada.

[2] Tilmann, C. P. (2002). “Emerging Aerodynamic Technologies for High-Altitude Long-Endurance SensorCraft UAVs.” *Air Force Research Lab Wright-Patterson AFB OH Air Vehicles Directorate*, Nov. 4.

[3] Darema, F. (2004). "Dynamic data driven applications systems: A new paradigm for application simulations and measurements." *Computational Science-ICCS 2004*: pp. 662-669.

[4] Fisher, M., Nocedal, J., Trémolet, Y., and Wright, S. J. (2009). "Data assimilation in weather forecasting: a case study in PDE-constrained optimization." *Optimization and Engineering* 10.3: pp. 409-426.

[5] Mandel, J., Bennethum, L., Beezley, J., Coen, J., Douglas, C., Kim, M., and Vodacek, A. (2008). “A wildland fire model with data assimilation.” *Math. Comput. Simul*. 79, pp. 584–606.

[6] Rodriguez, R., Cortés, A., and Margalef, T. (2009). “Injecting dynamic real-time data into a DDDAS for forest fire behavior prediction.” In: *Int. Conf. Comput. Sci. (ICCS),* vol. 5454, pp. 489–499. May 25, Springer, Berlin, Heidelberg.

[7] Akcelik, V., Biros, G., Draganescu, A., Ghattas, O., Hill, J., and Van Bloeman Waanders, B. (2005). “Dynamic data-driven inversion for terascale simulations:

real-time identification of airborne contaminants." *Proceedings of SC2005*, pp. 43–58. Nov. 12, Seattle, WA.

[8] Lieberman, C., Fidkowski, K.W., Van Bloemen Waanders, B. (2013). "Hessian-based model reduction: largescale inversion and prediction." *Int. J. Numer. Methods Fluids*, 71, pp. 135–150.

[9] Akcelik, V., Biros, G., Draganescu, A., Ghattas, O., and Hill, J. (2006). "Inversion of airborne contaminants in a regional model." In: *Int. Conf. Comput. Sci.* (ICCS),vol. 3993, pp. 481–488. May 28, Reading, UK.

[10] Madey, G. R., Blake, M. B., Poellabauer, C., Lu, H., McCune, R. R., and Wei, Y. (2012). "Applying DDDAS principles to command, control and mission planning for UAV swarms." *Procedia Computer Science*, 9: pp. 1177-1186.

[11] Peng, L., Doug, L., and Kamran, M. (2014). "Dynamic data driven application system for plume estimation using UAVs." *Journal of Intelligent & Robotic Systems*, 74.1-2: pp. 421-436.

[12] Khaleghi, A. M., Xu, D., Lobos, A., Minaeian, S., Son, Y.-J., and Liu, J. (2013). "Agent-based hardware-in-the-loop simulation for UAV/UGV surveillance and crowd control system." P*roceedings of the 2013 Winter Simulation Conference*: Simulation: Making Decisions in a Complex World. pp. 1455-1466, Dec. 8, IEEE Press.

[13] Uzkent, B., Hoffman, M. J., Vodacek, A., Kerekes, J.P., and Chen, B. (2013). "Feature matching and adaptive prediction models in an object tracking DDDAS." *Procedia Computer Science 18:* pp. 1939-1948. Jan. 1.

[14] Taieb, S. B., Hyndman, R. J., and Bontempi, G. (2014). "Machine learning strategies for multi-step-ahead time series forecasting," *Ph.D. Thesis*. Universit libre de Bruxelles, Belgium.

[15] Cheng, H.. Tan, P. N., Gao, J., and Scripps, J. (2006). "Multistep-ahead timeseries prediction." In W.K. Ng, M. Kitsuregawa, J. Li, and K. Chang (Eds.), PAKDD. Lecture notes in computer science (Vol. 3918, pp. 765–774). Springer. ISBN: 3-540-33206-5.

[16] Hamzacebi, C., Akay, D., and Kutay, F. (2009). "Comparison of direct and iterative artificial neural network forecast approaches in multi-periodic time series forecasting." *Expert Systems with Applications*, 36(2, Part 2), 3839–3844. ISSN: 0957-4174. <http://www.sciencedirect.com/science/article/B6V03-4S03RD5-6/2/8520e0674be6409b24eb1aca953bdb09>.

[17] Sorjamaa, A., Hao, J., Reyhani, N., Ji, Y., and Lendasse, A. (2007). "Methodology for longterm prediction of time series." *Neurocomputing*, 70(16-18), pp. 2861–2869.

[18] Tiao, G. C., and Tsay, R. S. (1994). "Some advances in non-linear and adaptive modelling in time-series." *Journal of Forecasting*, 13(2), pp. 109–131.

[19] Williams, R., and Zipser, D. (1989). "A learning algorithm for continually running fully recurrent neural networks." *Neural Computation 1,* pp. 270–280.

[20] McNames, J. (1998). "A nearest trajectory strategy for time series prediction." in: K.U. Leuven (ED.), Proceedings of the *International Workshop on Advanced Black- Box Techniques for Nonlinear Modeling*, Jul. 8, Belgium, 1998, pp. 112–128.

[21] Bontempi, G. (2008). "Long term time series prediction with multi-input multioutput local learning." *In Proceedings of the 2nd European symposium on time series prediction (TSP)*, ESTSP08, Helsinki, Finland (2018 Feb:pp. 145–154).

[22] Bontempi, G., and Taieb, S. B. (2011). "Conditionally dependent strategies for multiple-step-ahead prediction in local learning." *International Journal of Forecasting*, 27(3), pp. 689–699.

[23] Kline, D. M. (2004). "Methods for multi-step time series forecasting with neural networks." In G. P. Zhang (Ed.), *Neural networks in business forecasting,* pp. 226–250. Information Science Publishing.

[24] Taieb, S. B., Bontempi, G., Atiya, A. F., and Sorjamaa, A. (2012). "A review and comparison of strategies for multistep ahead time series forecasting based on the NN5 forecasting competition," *Expert Syst. Appl.*, vol. 39, no. 8, pp. 7067–7083, Jun. 2012.

[25] Sorjamaa, A., and Lendasse, A. (2006). "Time series prediction using DirRec strategy," in *Proc. Eur. Symp. Artif. Neural Netw.*, Bruges, pp. 143–148.

[26] Taieb, S. B., Bontempi, G., Atiya, A. F., and Sorjamaa, A. (2012). "A review and comparison of strategies for multistep ahead time series forecasting based on the NN5 forecasting competition," *Expert Syst. Appl.*, vol. 39, no. 8, pp. 7067–7083, Jun.

[27] Taieb, S. B., Bontempi, G., Sorjamaa, A., and Lendasse, A. (2009). "Long-term prediction of time series by combining direct and MIMO strategies," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, Atlanta, USA, Jun. 14, pp. 3054–3061.

[28] Taieb, S. B., Sorjamaa, A., and Bontempi, G. (2010). "Multiple-output modeling for multistep-ahead time series forecasting," *J. Neurocomput.*, vol. 73, pp. 1950–1957.

[29] Bao, Y., Xiong, T., and Hu, Z. (2014). "PSO-MISMO modeling strategy for multistep-ahead time series prediction." *IEEE transactions on cybernetics*, *44*(5), pp. 655-668.

[30] Amabili, M., Sarkar, A., and Paidoussis, M.P. (2003). "Reduced-order models for nonlinear vibrations of cylindrical shells via the proper orthogonal decomposition method." *J. Fluids Struct*. 18, pp. 227–250.

[31] Graham, M., and Kevrekidis, I. (1996). "Alternative approaches to the Karhunen-Loève decomposition for model reduction and data analysis." *Comput. Chem. Eng*. 20, pp. 495–506.

[32] Moradkhani, H., Hsu, K. L., Gupta, H., and Sorooshian, S. (2005). "Uncertainty assessment of hydrologic model states and parameters: Sequential data assimilation using the particle filter." *Water Resources Research*, 41, 5.

[33] Bernardo, J. M., Berger, J. O., Dawid, A. P., and Smith, A. F. M. (1992). "*Bayesian Statistics* 4." *Oxford University Press.* pp. 345–363, Valencia.

[34] Williams, C. K. I. (1998). "Prediction with Gaussian processes: From linear regression to linear prediction and beyond." *Natoasi series d behavioral and social sciences 89*: pp. 599-621.

[35] Thimmisetty, C., Ghanem, R., White, J., and Chen, X. (2017). "High-dimensional intrinsic interpolation using Gaussian process regression and diffusion maps." *Mathematical Geosciences*, Vol. 50, pp. 77-96.

[36] Thimmisetty, C., Aminzadeh, F., Rose, K., and Ghanem, R. (2018). "Multiscale Stochastic Representations using Polynomial Chaos Expansions with Gaussian Process Coefficients." *Data-Enabled Discovery and Applications*, Vol. 2, No.3.

[37] Kalman, R. (1960). "New approach to linear filtering and prediction problems." *J. Basic Eng.,* 82D, pp. 35–45.

[38] Jazwinsk, A. H. (1970). "Stochastic Processes and Filtering Theory." *Elsevier*, New York. pp. 376.

[39] Reichle, R. H., McLaughlin, D. B., and Entekhabi, D. (2002). "Hydrologic data assimilation with the ensemble Kalman filter." *Mon. Weather Rev*., 130, pp. 103–114.

[40] Bras, R. L., and Rodriguez-Iturbe, I. (1985). "Random Functions and Hydrology." *Addison-Wesley, Boston, Mass*. pp. 559.

[41] Evensen, G. (1994). "Sequential data assimilation with a nonlinear quasigeostrophic model using Monte Carlo methods to forecast error statistics." *J. Geophys. Res.,* 99, pp. 10,143–10,162.

[42] Burgers, G., van Leeuwen, P. J., and Evensen, G. (1998). "Analysis scheme in the ensemble Kalman filter." *Mon. Weather Rev.*, 126, pp. 1719–1724.

[43] Van Leeuwen, P. J. (1999). "Comment on data assimilation using an ensemble Kalman filter technique." *Mon. Weather Rev.,* 127, pp. 1374–1377.

[44] Kitagawa, G. (1996). "Monte Carlo filter and smoother for non-Gaussian non-linear state space models." *Journal of Computational and Graphical Statistics,* 5, pp. 1-25.

[45] Douc, R., Cappe, O., and Moulines, E. (2005). "Comparison of resampling schemes for particle filtering." *In 4th International Symposium on Image and Signal Processing and Analysis (ISPA*), Sept. 15, pp. 64-69.

[46] Moradkhani, H., Sorooshian, S., Gupta, H. V., and Houser, P. R. (2005a). "Dual state-parameter estimation of hydrological models using ensemble Kalman filter." *Adv. Water Resour.* 28(2), pp. 135–147.

[47] Kania, R., Kebbie-Anthony, A.B., Zhao, X., Azarm, S., and Balachandran, B. (2017). "Dynamic data-driven approach for unmanned aircraft systems." To appear, *Handbook of Dynamic Data Driven Application System*, Springer Nature Swithzerland AG 2018.

[48] Zhao, X., Kania, R., Kebbie-Anthony, A.B., Azarm, S., and Balachandran, B. (2018). "Dynamic Data-Driven Aeroelastic Response Prediction with Discrete Sensor Observations," In *2018 AIAA Non-Deterministic Approaches Conference*, pp. 2173. Jan. 8-12, Kissimmee, Florida.

[49] Kebbie-Anthony, A.B., Gumerov, N., Preidikman, S., Balachandran, B., and Azarm, S. (2018). "Fast Multipole Method for Nonlinear, Unsteady Aerodynamic Simulations," In *2018 AIAA Modeling and Simulation Technologies Conference*, pp. 1929. Jan. 8-12, Kissimmee, Florida.

[50] Roccia, B., Preidikman, S., and Balachandran, B. (2017). "Computational dynamics of flapping wings in hover flight: A co-simulation strategy," *AIAA Journal*, vol. 55(6), pp. 1806-1822.

[51] Hsu, S. T.-S., Fitzgerald, T., Nguyen, V., Patel, T., and Balachandran, B. (2017). "Motion visualization and estimation for flapping wing systems," *Acta Mechanica Sinica*, 33(2), pp. 1806-1822.

[52] Wei, X. and Du, X. (2019). "Uncertainty analysis for time-and space-dependent responses with random variables." *Journal of Mechanical Design*, 141(2), 021402.

[53] Xi, Z., Youn, B.D., and Hu, C. (2010). "Random field characterization considering statistical dependence for probability analysis and design." *Journal of Mechanical Design*, 132(10), 101008.

[54] Beek, A. V., Li, M., and Ren, C. (2018). "Heuristics-enhanced model fusion considering incomplete data using kriging models." *Journal of Mechanical Design*, 140(2), 021403.

[55] Yin, X., Lee, S., Chen, W., Liu, W.K., and Horstemeyer, M.F. (2009). "Efficient random field uncertainty propagation in design using multiscale analysis." *Journal of Mechanical Design*, 131(2), 021006.

[56] Peddada, S. R., Tannous, P. J., Alleyne, A. G., and Allison, J. T. (2020). "Optimal sensor placement methods in active high power density electronic systems with experimental validation." *Journal of Mechanical Design*, 142(2), 023501.

[57] Box, G. E., and Pierce, D. A. (1970). "Distribution of residual autocorrelations in autoregressive-integrated moving average time series models." *Journal of the American statistical Association*, *65*(332), 1509-1526.

[58] Fu, R., Zhang, Z., and Li, L. (2016). "Using LSTM and GRU neural network methods for traffic flow prediction." *In 2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC),* IEEE, 324-328.

[59] Min, W., and Laura, W. (2011). "Real-time road traffic prediction with spatio-temporal correlations." *Transportation Research Part C*: Emerging Technologies 19, no. 4: 606-616.

[60] Quan, H., Srinivasan, D., and Khosravi, A. (2014). "Short-term load and wind power forecasting using neural network-based prediction intervals." *IEEE transactions on neural networks and learning systems*, *25*(2), 303-315.

[61] Dering, M. L. and Tucker, C. S. (2017). "A convolutional neural network model for predicting a product's function, given its form." *Journal of Mechanical Design*, 139(11), 111408.

[62] Das, M. and Ghosh, S. K. (2019). "FB-STEP: a fuzzy Bayesian network based data-driven framework for spatio-temporal prediction of climatological time series data." *Expert Systems with Applications*, 117, 211-227.

[63] Cheng, C., Sa-Ngasoongsong, A., Beyca, O., Le, T., Yang, H., Kong, Z., and, Bukkapatnam, S. T. (2015). "Time series forecasting for nonlinear and non-stationary processes: A review and comparative study." *IIE Transactions*, 47(10), 1053-1071.

[64] Sabbioni, E., Bao, R., Cheli, F., and Tarsitano, D. (2017). "A particle filter approach for identifying tire model parameters from full-scale experimental tests." *Journal of Mechanical Design*, 139(2), 021403.

[65] Rasmussen, C. E. and Williams, C. K. I. (2006). "Gaussian process for machine learning." *MIT Press,* Cambridge, MA.

[66] Bernardo, J. M., Berger, J. O., Dawid, A. P., and Smith, A. F. M. (1992). "Bayesian Statistics 4." *Oxford University Press*. 345–363, Valencia.

[67] Li, M. and Wang, Z. (2018). "Confidence-driven design optimization using Gaussian process metamodeling with insufficient data." *Journal of Mechanical Design* 140(12), 121405.

[68] Bracewell, R. N. (1986). "The Fourier transform and its applications." *New York: McGraw-Hill,* Vol. 31999.

[69] Daubechies, I. (1990). "The wavelet transform, time-frequency localization and signal analysis." *IEEE Transactions on Information Theory*, 36(5), 961-1005.

[70] Youn, B.D., Xi, Z., and Wang, P. (2008). "Eigenvector dimension reduction (EDR) method for sensitivity-free probability analysis." Structural and Multidisciplinary Optimization, 37(1), 13-28.

[71] Zhao, X., Kebbie-Anthony, A.B., Azarm, S., and Balachandran, B. (2019). "Dynamic data-driven multi-step-ahead prediction with simulation data and sensor measurment data." *AIAA Journal*, 1-10.

[72] Lathauwer, L. D., Moor, B. D., and Vandewalle, J. (2000). "A multilinear singular value decomposition." *SIAM journal on Matrix Analysis and Applications,* 21(4), 1253-1278.

[73] Vasilescu, M. A. O. and Terzopoulos, D. (2007). "Multilinear projection for appearance-based recognition in the tensor framework." *In Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference*. IEEE, 1-8.

[74] Fanaee-T, H., and Gama, J. (2015). "Eigenevent: an algorithm for event detection from complex data streams in syndromic surveillance." *Intelligent Data Analysis*, 19(3), 597-616.

[75] Stewart, G. W. (1993). "On the early history of the singular value decomposition." *SIAM review,* 35(4), 551-566.

[76] Kebbie-Anthony, A.B., Gumerov, N.A., Preidikman, S., Balachandran, B., and Azarm, S. (2019). "Fast multipole accelerated unsteady vortex lattice method based computations," *Journal of Aerospace Information System*, 11(6), 237-248.

[77] Wang, G., Riaz, A., and Balachandran, B. (2021). "Continuum Modeling and Simulation of Robotic Appendage Interaction With Granular Material." *Journal of Applied Mechanics*, 88, p. 021013.

[78] Li, C., Umbanhowar, P.B., Komsuoglu, H., Koditschek, D.E., Goldman, D.I., 2009. Sensitive dependence of the motion of a legged robot on granular media. Proceedings of the National Academy of Sciences 106, 3029–3034.

[79] Mayeres, I., Ochelen, S., and Proost, S. (1996) "The marginal external costs of urban transport." *Transportation Research Part D: Transport and Environment,* vol. 1, no. 2, pp. 111-130.

[80] Miandoabchi, E., Daneshzand, F., Szeto, W.Y., and Farahani, R.Z. (2013) "Multi-objective discrete urban road network design." *Computers & Operations Research,* vol. 40, no. 10, pp. 2429-2449.

[81] Long, J., Szeto, W.Y., and Huang, H.J. (2014) "A bi-objective turning restriction design problem in urban road networks." *European Journal of Operational Research,* vol. 237, no. 2, pp. 426-439.

[82] Poorzahedy, H., and Abulghasemi, F. (2005) "Application of ant system to network design problem." *Transportation,* vol. 32, no. 3, pp. 251-273.

[83] Cong, Z., Schutter, B.D., and Babuška, R. (2015) "Co-design of traffic network topology and control measures." *Transportation Research Part C: Emerging Technologies*, vol. 54, pp. 56-73.

[84] Balaji, P. G., German, X., and Srinivasan, D. (2010) "Urban traffic signal control using reinforcement learning agents." *IET Intelligent Transport Systems,* vol. 4, no. 3, pp. 177-188.

[85] Arel, I., Liu C., Urbanik, T., and Kohls, A.G. (2010) "Reinforcement learning-based multi-agent system for network traffic signal control." *IET Intelligent Transport Systems,* vol. 4, no. 2, pp. 128-135.

[86] Guo, M., Wang P, Chan, C.Y., and Askary, S. (2019) "A reinforcement learning approach for intelligent traffic signal control at urban intersections." *In 2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, IEEE, Auckland, New Zealand, Oct. 27-30, 2019, pp. 4242-4247.

[87] Li L., Lv Y., and Wang, F.Y. (2016) "Traffic signal timing via deep reinforcement learning." *IEEE/CAA Journal of Automatica Sinica,* vol. 3, no. 3, pp. 247–254.

[88] Wei H., Zheng, G., Yao, H., and Li, Z. (2018) "IntelliLight: a reinforcement learning approach for intelligent traffic light control." *In ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*, New York, NY, July 2019, pp. 2496–2505.

[89] Wade, G., and Razavi, S. (2019) "Asynchronous n-step Q-learning adaptive traffic signal control." *Procedia Computer Science,* vol. 23, no. 130, pp. 26-33.

[90] Elise, V.P., and Oliehoek, F.A. (2016) "Coordinated deep reinforcement learners for traffic light control." *Proceedings of Learning, Inference and Control of Multi-Agent Systems* (at NIPS 2016).

[91] Gao, J., Shen, Y., Ito, M., and Shiratori, N. (2018) "Bias Based General Framework for Delay Reduction in Backpressure Routing Algorithm." *In 2018 International Conference on Computing, Networking and Communications (ICNC)*, IEEE, Maui, Hawaii, USA, March 5-8, pp. 215-219.

[92] El-Tantawy, S., Abdulhai, B., and Abdelgawad, H. (2013) "Multiagent reinforcement learning for integrated network of adaptive traffic signal controllers (MARLIN-ATSC): methodology and large-scale application on downtown Toronto." *Intelligent Transportation Systems (ITS),* vol. 14, no. 3, pp. 1140–1150.

[93] Chin, Y. K., Bolong, N., Kiring, A., Yang, S.S., and Teo, K.T.K. (2011) "Q-learning based traffic optimization in management of signal timing plan," *International Journal of Simulation, Systems, Science & Technology*, vol. 12, no. 3, pp. 29–35.

[94] Abdoos, M., Mozayani, N., and Bazzan, A.L. (2013) "Holonic multi-agent system for traffic signals control." *Engineering Applications of Artificial Intelligence,* vol. 26, no. 5, pp. 1575–1587.

[95] Prashanth, L. A., and Bhatnagar, S. (2010) "Reinforcement learning with function approximation for traffic signal control." *IEEE Transactions on Intelligent Transportation Systems,* vol. *12*, no. 2, pp. 412-421.

[96] Liang X., Du, X., Wang, G., and Han, Z. (2019) "A deep reinforcement learning network for traffic light cycle control." *IEEE Transactions on Vehicular Technology,* vol. 68, no.2, pp. 1243-1253.

[97] Mousavi, S.S., Schukat, M., and Howley, E. (2017) "Traffic light control using deep policy-gradient and value-function-based reinforcement learning." *Intelligent Transport Systems (ITS)*, vol. 11, no. 7, pp. 417–423.

[98] Casas, N. (2017) "Deep deterministic policy gradient for urban traffic light control." arXiv preprint arXiv:1703.09035, March, 27.

[99] Tian, T., Bao, F., Deng, Y., Jin, A., Dai, Q., and Wang, J. (2019) "Cooperative deep reinforcement learning for large-scale traffic grid signal control." *IEEE transactions on cybernetics*, vol 50, no. 6, pp. 2687-2700.

[100] Castro da Silva, A.B., Oliveria, D., and Basso, EW. (2006) "Adaptive traffic control with reinforcement learning." In Conference on Autonomous Agents and Multiagent Systems (AAMAS), Singapore, Singapore, May 2016, pp. 80–86.

[101] Chu, T., Wang, J., Codecà, L., and Li, A. (2019)"Multi-agent deep reinforcement learning for large-scale traffic signal control." IEEE Transactions on Intelligent Transportation Systems, vol. 21, no. 3, pp. 1086-1095.

[102] Khamis, M.A., and Gomaa, W. (2014) "Adaptive multi-objective reinforcement learning with hybrid exploration for traffic signal control based on cooperative multi-agent framework." Engineering Applications of Artificial Intelligence, vol. 29, pp. 134-151.

[103] Devailly, F.X., Larocque, D., and Charlin, L. (2021) “Ig-rl: Inductive graph reinforcement learning for massive-scale traffic signal control.” In IEEE Transactions on Intelligent Transportation Systems, pp. 1-12.

[104] Puterman, M. L. (1990) “Markov decision processes.” in Handbooks in Operations Research and Management Science, vol. 2. Chichester, U.K.: Wiley, pp. 331–434.

[105] Luck, K.S., Amor ,H.B., and Calandra, R. (2020) “Data-efficient co-adaptation of morphology and behaviour with deep reinforcement learning.” In Conference on Robot Learning, PMLR, vol. 100, pp. 854-869. May, 2020.

[106] Schaff, C., Yunis, D., Chakrabarti, A., and Walter, M.R.. (2013) “Jointly learning to construct and control agents using deep reinforcement learning.” In 2019 International Conference on Robotics and Automation (ICRA), IEEE, Montreal, QC, Canada, May 20-24, 2019, pp. 9798–9805.

[107] Ha, D. (2018) “Reinforcement learning for improving agent design.” Artificial Life, vol. 25, no. 4, pp. 352-365.

[108] Williams, R.J. (1992) “Simple statistical gradient-following algorithms for connectionist reinforcement learning.” Machine learning, vol. 8, no. 3-4, pp. 229–256.

[109] Micha, S. (1981) “A strong-connectivity algorithm and its applications to data flow analysis.” Computers and Mathematics with Applications, vol. 7, no.1, pp.67–72.

[110] Van Otterlo, M., and Wiering, M. (2012) "Reinforcement learning and markov decision processes." In Reinforcement Learning, Springer, Berlin, Heidelberg. pp. 3-42.

[111] Bellman, R., and Kalaba, R. (1957) "Dynamic programming and statistical communication theory." Proceedings of the National Academy of Sciences of the United States of America, vol. 43, no. 8, pp. 749.

[112] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., et al. (2015) "Human-level control through deep reinforcement learning." Nature, vol. 518, no. 7540, pp. 529-533.

[113] Cormen, T.H., Charles E.L., Ronald L.R., and Clifford S. (2009) "Introduction to algorithms." MIT press, July 31.

[114] Mavrommati, A. (Dec. 2020). Mathworks/OpenTrafficLab: 1.0.0. [Online]. Available: https://doi.org/10.5281/zenodo.4354100

[115] MATLAB^R, Release R2020b, MathWorksR , Natick, MA, USA, Sep. 2020.

[116] Automated Driving ToolboxTM, Release R2020b, MathWorksR , Natick, MA, USA, Sep. 2020.

[117] Hirvensalo, M. (2013). "Quantum computing." Springer Science & Business Media. March 14.