LITERATURE REVIEW

Learning a reduced-order dynamic model for quadruped robots

Author: Gioele Buriani

Supervisor: Cosimo Della Santina

April 5, 2023

Abstract

Learning a reduced-order dynamic model for quadruped robots

by Gioele BURIANI

In recent years, quadruped robots have gained increasing popularity, both in research and popular culture. Model-based control algorithms are commonly used to control modern quadrupeds. However, the requirement of a system model can be a limitation, as models may not be available or may be too complex and computationally expensive for real-time control. This literature review investigates the potential for generating simplified yet efficient models for quadruped robots using machine learning techniques. Specifically, the review outlines the general characteristics of legged robots and quadrupeds, explores different machine-learning techniques for generating models of complex systems, and investigates the state of the art in applying machine learning to generate models for quadruped robots. This review reveals a significant gap in the application of model learning to the case of quadruped robots. By addressing this gap, the discussion aims to identify opportunities for future research on the topic.

Contents

Chapter 1

Goal of the review and methodology

This document aims to provide a comprehensive review of the current state of the art in the area of model learning for complex systems, with a specific focus on quadruped robots.

In recent years, quadruped robots have gained significant importance due to their diverse applications in fields such as agriculture, exploration, rescue missions, and entertainment, which often involve social interactions with humans. To enable efficient control of such robots, model-based control methods have become increasingly popular. However, the success of such methods hinges on the availability of reliable and accurate models of quadruped robots. The purpose of this study is thus to explore various techniques that can be used to generate such models from data, while also identifying any research gaps that can be addressed by my future studies.

The paper begins by highlighting the key characteristics of legged robots, including their advantages that have contributed to their growing importance in recent times. The document then delves into quadruped robots, discussing their background and relevant dynamical characteristics for this review. The next chapter provides an in-depth analysis of various machine learning techniques that can be applied to learn models of complex systems, ranging from traditional linear regression methods to more advanced approaches based on deep learning and evolutionary algorithms. In the following chapter, the previous two are integrated to examine the state of the art in model learning techniques applied specifically to quadruped robots. Finally, the paper concludes by summarizing the findings and identifying any gaps in the current research, which will serve as the basis for formulating the problem statement for my future thesis work.

1.1 Literature review methodology

The method employed for researching articles in the study was designed according to the guidelines provided by PRISMA [1].

1.1.1 Eligibility criteria

The eligibility criteria for the sources of the review followed three main categories: mechanical aspect, modelling methodology and publication criteria.

The mechanical aspect pertains to the characteristics of the subject of the source. Depending on the chapter, the requested characteristics were: being a legged robot (Chapters 1 and 3), being a quadruped robot (Chapters 1 and 3), being a non-linear dynamical system (Chapter 2) or being a hybrid system (Chapter 3). Depending on the specific section taken into consideration, all sources not adhering to these characteristics were discarded.

The modelling methodology considers the way in which a source addresses the generation of a model. Depending on the chapter, the requested characteristics were: using reduced-order template models (Chapter 2), using regression techniques including linear regression, support vector regression, decision tree regression and gaussian regression (Chapters 3 and 4), using methods based on deep learning including recurrent neural networks, transformers or autoencoders (Chapters 3 and 4) or using equation discovery techniques such as genetic programming, symbolic regression or sparse identification of nonlinear dynamics (Chapters 3 and 4).

Lastly, the publication criteria were based on the formal characteristics of the source. Eligible sources included digital journals, articles, conference proceedings, review papers, lectures, and books, while patents were excluded. Additionally, sources were ranked according to the number of citations obtained per year, with sources receiving less than 10 citations per year only included if published after 2022 or if they demonstrated significant relevance to the topic.

1.1.2 Information sources

The Google Scholar search engine served as the main source of information for this review. Journals or conference proceedings outside of this database were not considered, nor were physical copies of non-digitized papers. Further references were obtained by manually searching through the citations of articles found in Google Scholar.

To ensure a formal academic tone in the language used in the review, the OpenAI software ChatGPT was utilized for rephrasing the text.

Furthermore, to ensure the grammatical accuracy of the entire paper, the Grammarly software was employed.

1.1.3 Search strategy

The initial search query utilized to obtain the results is presented as follows:

["legged locomotion" OR "quadruped robots" OR "quadruped dynamics] AND ["model learning" OR "learning dynamic model" OR "modelling non-linear system"] AND ["model learning hybrid system" OR "model learning quadruped" OR "model learning legged"]

To narrow down the search, the most highly cited papers were given priority, with review papers being favoured. The abstracts of these highly cited papers were analyzed first, and the most relevant papers were selected for more in-depth reading. Additional papers were retrieved from the reference lists and citations of the selected papers. This iterative process was repeated in a hierarchical manner until enough papers were obtained to comprehensively cover the relevant topics.

1.2 Report structure

The literature review is structured as follows:

- **Chapter 2** provides an introduction to the context of the review, which is focused on quadruped robots. It begins by outlining the main characteristics of legged systems and emphasizing their superiority over wheeled ones for certain applications. Then, it narrows the focus to quadrupedal robots and provides a brief historical overview of the field, followed by a discussion of relevant dynamics features, such as their floating base dynamics and reduced-order template models.

- **Chapter 3** focuses on the main topic of the review, which is the analysis of common model learning techniques. It first covers standard regression techniques, such as linear regression, support vector regression, decision tree regression, and Gaussian regression. The chapter then moves on to deep learning tools, discussing recurrent neural networks, dimensionality reduction methods using autoencoders, and innovative transformers. Finally, equation discovery methods based on evolutionary algorithms are analyzed, including genetic programming, its application in the context of symbolic regression, and the more recent sparse identification of nonlinear dynamics.

- **Chapter 4** combines the previous two chapters and concentrates on model learning techniques specifically applied to legged robots. It first analyzes hybrid systems in general and introduces the hybrid zero dynamics model, as well as machine learning methods used to learn these types of models. The chapter then focuses specifically on quadrupeds, highlighting studies on parameter identification methods and learning-based approaches to model quadruped robots.

- **Chapter 5** provides a discussion of the most significant findings of the literature review, identifies gaps in the existing research, and concludes with the problem statement for the thesis.

Chapter 2

Introduction to quadrupeds

2.1 Legged locomotion

To comprehend the essence of legged locomotion, it is essential to first comprehend its definition. Locomotion can be defined as the process of movement from one place to another, which can be accomplished through various means. In particular, this paper focuses on the movement achieved through the use of legs, intended as appendages attached to the body for the purpose of support and movement.

The notion of legged locomotion can be simplified as a reflection of the manner in which humans usually move, known as bipedal locomotion. However, not all organisms move through the use of only two legs; many mammals, for instance, utilize four legs in a movement known as quadrupedal locomotion. Other organisms, mainly insects and arachnids, move with the use of six or more legs. Despite the differences among these organisms, they share significant similarities.

The most crucial aspect of legged locomotion, relevant to this paper, is the capability of using discrete footholds for each foot [2]. This means that organisms that move through the use of legs can place each foot on separate and distinct areas during locomotion. This feature is paramount for the purpose of walking on irregular terrain, which explains why many organisms have evolved to move in this manner. In fact, the versatility of leg configuration enables adaptation to surface irregularities and the selection of specific ground points for contacts based on their conditions.

The capability of dynamic stability also constitutes a significant aspect of legged locomotion, as highlighted by [3]. This refers to the momentary displacement of an organism's centre of mass outside of its feet's support polygon in order to facilitate longer strides and utilize gravitational energy for movement. Additionally, the potential for dynamic reconfiguration of the body's centre of mass enables high-power manipulation. The ability to modify the body's support structure allows for the alteration of the centre of mass without compromising balance, thereby generating momentum and increasing power output. This phenomenon can be observed in actions such as a baseball player's bat swing.

The development of legged robots, designed to mimic the advantages of legged organisms, has been informed by three distinct approaches, as detailed in [4]. The first approach is informed by neurobiology, with a focus on

the role of Central Pattern Generators (CPGs), which are networks of neurons located in the spinal cords of both vertebrates and invertebrates. These CPGs are capable of generating muscular activity in the absence of sensory feedback. Another approach, the reflex-driven approach, emphasizes the importance of proprioceptive feedback and coordination between and within limbs in the formation of locomotor patterns. Finally, biomechanical studies concentrate on the dynamics between the body and limbs in their environment, with a tendency to overlook the specifics of neural information. Through these three approaches, significant advancements have been made in replicating the movement patterns of legged organisms and have culminated in the current state-of-the-art in legged robots (Figure 2.1).

FIGURE 2.1: A quadruped (Spot) and biped (Atlas) legged robots by Boston Dynamics.

It is now significant to briefly discuss whether these legged robots actually bring concrete advantages to the field of technology when compared to the alternative means of ground locomotion.

2.1.1 Comparison with wheeled locomotion

The design of machines for ground locomotion has been primarily based on wheels for thousands of years, with research on legged robots only beginning in the last century. This highlights the first limitation of legged locomotion compared to wheeled locomotion, which is complexity. The recent advent of research in legged machines is attributed to the advancements in computing technology, which has allowed for the integration of small, yet powerful computers into small vehicles [5]. As stated in [3], while a car only requires two controllable degrees of freedom, a legged system necessitates at least three degrees of freedom per leg to effectively manage contact interactions in a three-dimensional environment. The complexity involved in controlling legged robots remains one of the primary disadvantages of legged systems

in comparison to wheeled systems, and hence, research in this field is still in its early stages.

The issue of stability presents a second challenge that exacerbates the complexity of control in legged systems. Wheeled vehicles with three or more wheels can maintain their balance without the need for additional control [6]. Conversely, legged beings that rely on dynamic stability, such as humans, must constantly regulate their balance. Consequently, the utilization of dynamic stability in legged systems [3] necessitates a higher degree of control complexity when compared to wheeled vehicles.

The third problem of legged vehicles when compared to wheeled ones is energy efficiency. In fact, legged locomotion requires nearly 100 times more power compared to wheeled locomotion on flat, hard surfaces, as can be seen in Figure 2.2, provided by [6]. This is also due to the increased complexity, explained previously: more degrees of freedom require more actuators which require more energy. This is the reason why a car or a train on a straight road can easily outperform any legged being both in terms of speed and durability. However, if this wheeled vehicle was to be put on soft ground, such as mud or sand, its performance would steeply deteriorate. This is also shown in Figure 2.2, where it can be noticed how legged locomotion outperforms wheeled one on soft ground. This is due to the fact that legs' discrete footholds deform the terrain less than wheeled vehicles and thus require less energy to get out of the depression [2].

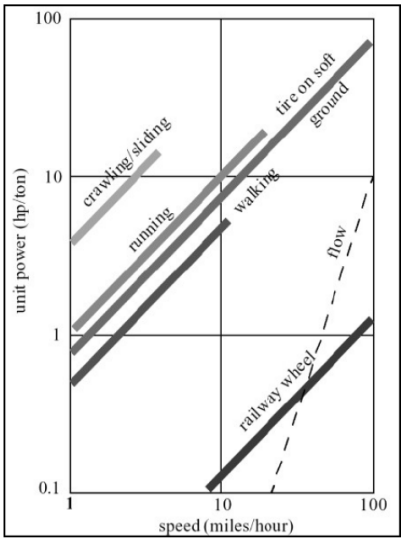

FIGURE 2.2: Power consumption of different locomotion mechanisms. Image from [6].

Despite the aforementioned challenges, the primary reason for the sustained interest in legged systems in the field of robotics is their ability to adapt to a variety of ground types, as discussed in Section 2.1. In fact, more than half of the earth's terrain is inaccessible to traditional wheeled vehicles due to the presence of obstacles and irregular surfaces. This is because wheeled vehicles require a continuous support surface, which is not always available, whereas legged robots only require small, discrete supports [2].

This versatility makes legged robots suitable for a wide range of applications that demand the ability to traverse uneven terrain, ranging from forest floors to staircases, which wheeled vehicles are unable to negotiate.

Figure 2.3, generated by [7], presents a table that succinctly summarizes the benefits and drawbacks of both wheeled and legged locomotion mechanisms. In conclusion, it is important to recognize that there is no definitive superiority of one locomotion mechanism over the other, as the performance of each is dependent on the specific operating environment. As a result, having versatile options for both wheeled and legged locomotion is critical to ensuring that a robot can be deployed in a wide range of scenarios and environments.

Technical Criteria	*Wheel Robot*	*Legged Robot*
Maneuverability	X	✓
Transvers ability	X	✓
controllability	✓	X
Terrain Land	X	✓
Efficiency	X	✓
Stability	✓	X
Cost effective	✓	X
Navigation over obstacles	X	✓

FIGURE 2.3: Table that compares wheeled and legged robots. Image from [7].

Leg-wheel hybrid robots

Given the unique advantages each mode of locomotion presents in different scenarios, certain researchers have explored ways to combine the benefits of both legged and wheeled locomotion into hybrid robots that utilize both legs and wheels. Examples of such leg-wheel hybrid robots can be found in [8].

The initial category of hybrid robots encompasses those equipped with legs and wheels. Each leg of these robots is fitted with a wheel at its tip, enabling them to benefit from the energy-efficient nature of wheels on flat surfaces and the versatility of legs in navigating uneven terrain. Illustrative examples of this type of hybrid robots are the Roller Walker [9], as developed by S. Hirose in 1996 (shown in Figure 2.4) and the recent ANYmal from ANYbotics when equipped with wheels [10].

An alternative approach involves the utilization of rotating legs, which aim to combine the energy efficiency of having a single rotational actuator per leg with the ability to navigate obstacles through discrete contact surfaces. Representative examples of this concept include the RHex, developed by M. Buehler [11], and the Whegs II, created by T.J. Allen [12] (illustrated in Figure 2.5).

After describing legged locomotion in general, the paper will now focus more specifically on quadrupeds.

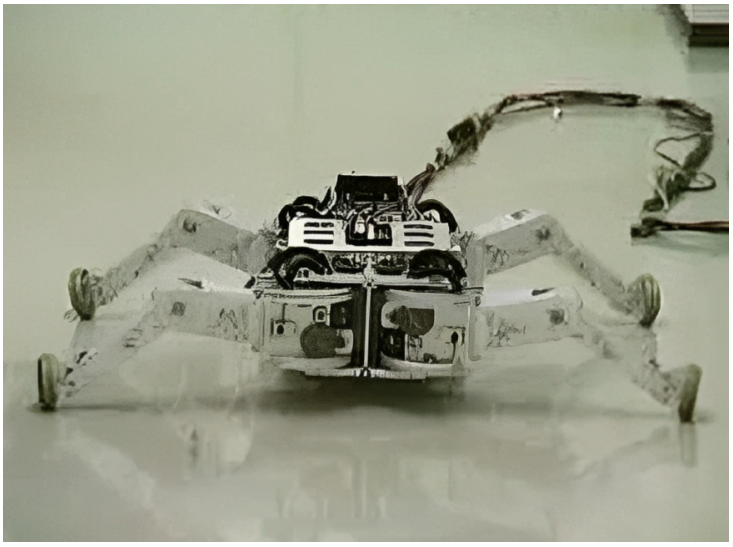

FIGURE 2.4: Roller Walker by S. Hirose.

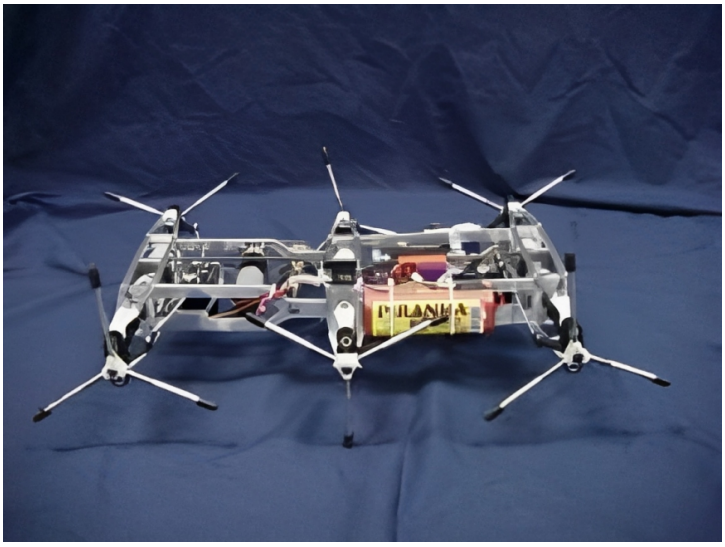

FIGURE 2.5: Whegs II by T.J. Allen.

2.2 Quadrupeds

The preceding segment succinctly highlighted certain attributes of legged robots, ranging from bipeds to hexapods. Among these various forms of legged robots, quadrupeds have been deemed as the optimal option in terms of mobility, stability during locomotion, and computational complexity [7]. In fact, quadrupeds exhibit both higher speed and stability compared to bipeds, while being less complex to control than hexapods. This has resulted in substantial research focus on this particular aspect of legged robots, and accordingly, justifies the scope of the present research endeavour.

The next subsection will mention both historical and more modern relevant quadrupeds.

2.2.1 History and modern quadrupeds

The present subsection will provide a brief overview of the historical background of quadruped robots, synthesizing information from various sources, with a primary emphasis on [7].

Historical background

The concept of the "first quadruped robot" doesn't really have a clear and univocal answer. It is possible, however, to track the invention of the first machine walking on four legs back to 1893 when L. A. Rygg created his human-powered mechanical horse (Figure 2.6). This machine is very far from what is considered a quadruped robot today, but it was a necessary start for the progress of the field.

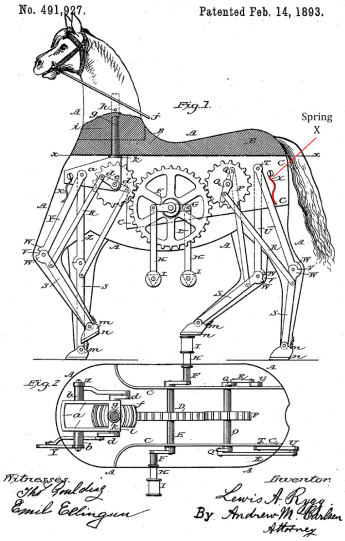

FIGURE 2.6: Mechanical horse by L.A. Rygg.

Following this seminal invention, additional quadrupedal machines controlled by human operators were developed. One notable instance is the Walking Truck [13], which was created in 1968 by R. Mosher, building upon prior work by A.C. Hutchinson. This machine was capable of functioning with legs that were independently controlled, but still manually operated.

In the same year, at the University of Southern California, the creation of the first quadruped robot with autonomous control was achieved. This groundbreaking innovation was referred to as Phony Pony [14] or "The California Horse" (as depicted in Figure 2.7). The robot was equipped with two rotary joints per leg and electric actuation, allowing for the autonomous generation of diverse gait patterns, including the ability to crawl.

Globally renowned institutions invested in quadrupedal robot research, but the Tokyo Institute of Technology, under the direction of S. Hirose, made a seminal contribution to the field [15]. The researchers of the institute designed their first quadrupedal robots by getting inspiration from spiders, which resulted in 1979 in the creation of PV-II (depicted in Figure 2.8). This

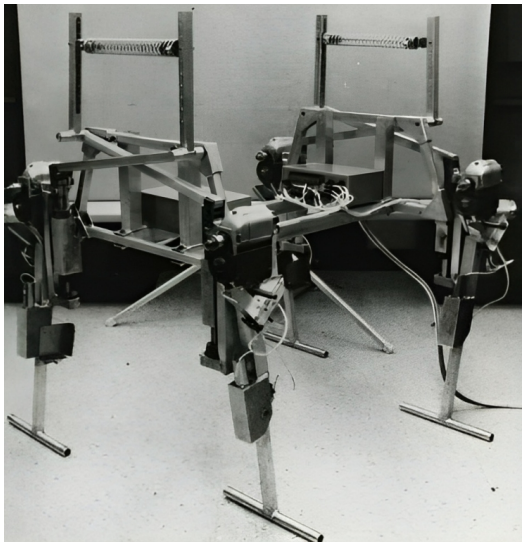

FIGURE 2.7: Phony Pony from the University of Southern California.

innovation marked the first instance of a legged robot capable of ascending stairs through sensor-based motion control.

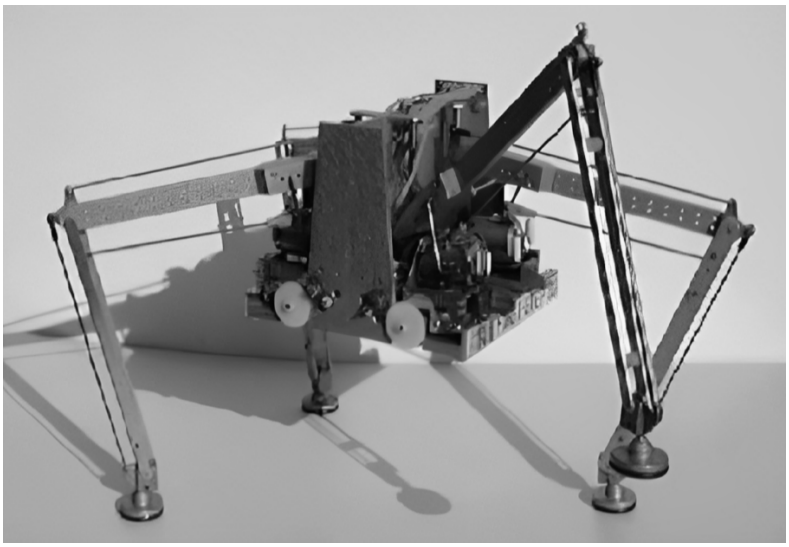

FIGURE 2.8: PV-II by S. Hirose.

All previously mentioned robots are based on the principle of static stability, which requires maintaining the centre of mass within the convex polygon formed by the robot's legs in contact with the ground. While this design approach ensured stability while traversing flat terrain, it also limited the robot's speed and agility. However, in 1986, M. H. Raibert made a crucial advancement in the field with the publication of his book "Legged Robots that Balance" [16]. In the book, Raibert introduced the concept of dynamic stability, as detailed in Section 2.1. This concept played a critical role in the development of quadruped robots with increased speed and agility in the following years.

The field of robotics has greatly benefited from the contributions made by researchers affiliated with Japanese institutions. The most notable and widely recognized achievements include the thirty-year-long TITAN series by S. Hirose [17] (Figure 2.9), the Collie series by H. Miura [18], and the TEKKEN series by H. Kimura [19], which was inspired by his earlier robot Patrush-I [20]. It is important to note that the TEKKEN series, released in 2004, represented a significant advance in the field by incorporating a neural system and a Central Pattern Generator (CPG) to control the robots, thus serving as a major influence on subsequent work in the field.

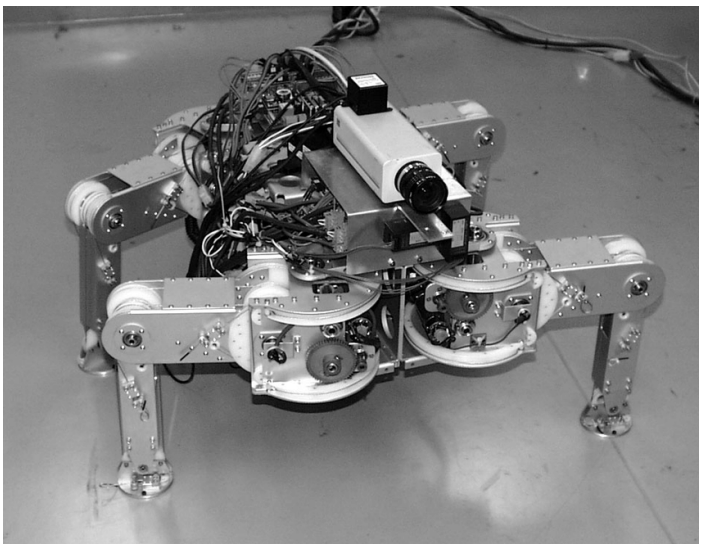

FIGURE 2.9: TITAN-VIII by S. Hirose.

Together with Japan, institutions from all around the world contributed to the research on quadruped robots.

Prior to the establishment of Boston Dynamics, the most notable accomplishment in the field of quadrupedal robots in the United States can be attributed to M. Buehler and his design of the SCOUT [21] (Figure 2.10). The SCOUT, a relatively simple quadruped, demonstrated capabilities in walking, running, and ascending stairs. In 2004, a collaborative effort between Ohio State University and Stamford University resulted in the creation of the KOLT quadruped robot [22]. The development of this quadruped was aimed at investigating the gallop gait of a cursorial quadruped with the aid of Dynamic Articulated Structure for High Performance (DASH) legs.

Europe has also made significant contributions to the field of quadrupedal robots, particularly in Germany, with the development of the BISAM walking machine [23] at the Karlsruhe Institute of Technology. Spain has also made notable contributions with the design of the SILO4 quadruped robot [24] (Figure 2.11) by the Industrial Automation Institute (CSIC). Both of these robots have demonstrated exceptional robustness, adaptability to various terrains, and stability in their respective designs.

Additionally, universities in China have also made significant contributions to the research and development of quadruped robots. In 2010, Shandong University introduced the SCalf quadruped robot [25] (Figure 2.12),

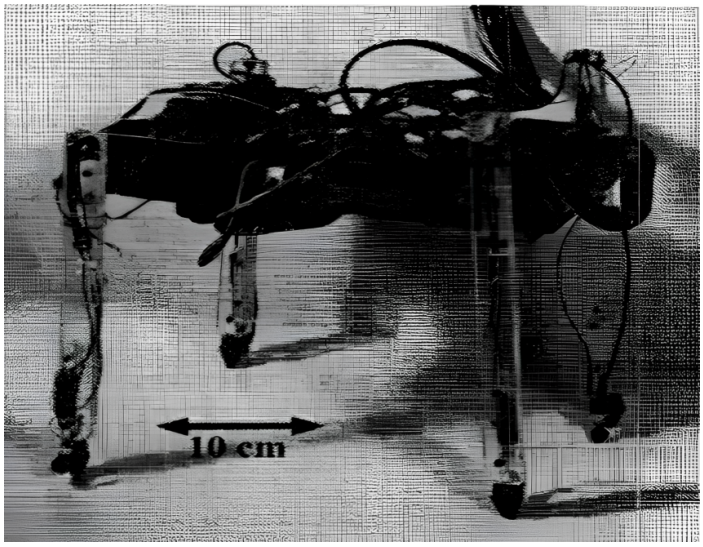

FIGURE 2.10: SCOUT by M. Buehler.

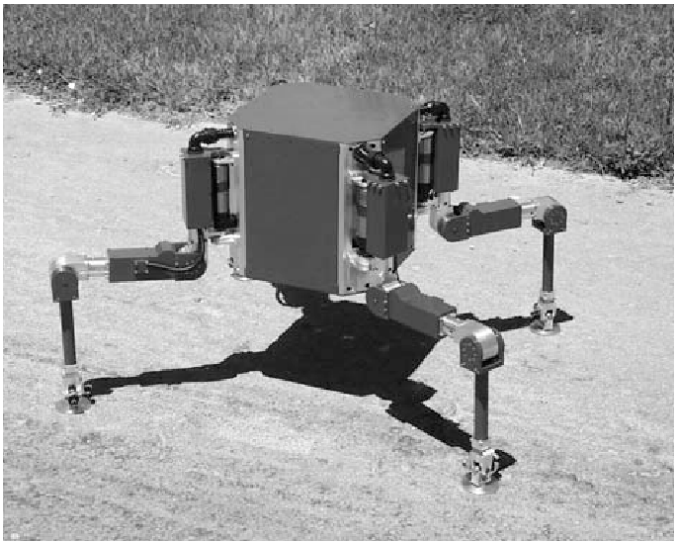

FIGURE 2.11: SILO4 from CSIC.

which was powered by an IC engine and had 12 degrees of freedom. Four years later, the Harbin Institute of Technology designed and developed another noteworthy quadruped called MBBOT [26], where all actuators were powered by an external hydraulic pump. These contributions from Chinese universities have resulted in significant advancements in the field.

The aforementioned machines represent crucial advancements in the evolution of contemporary quadrupedal robots. This section aimed to provide a comprehensive understanding of the significant history and effort that has been invested in the development of these sophisticated machines. The subsequent subsection will briefly examine the more recent and widely recognized quadrupedal robots.

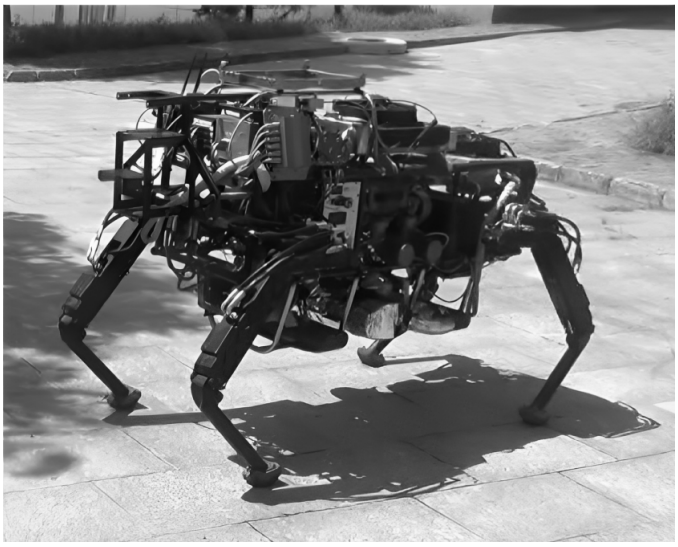

FIGURE 2.12: SCalf from Shandong University.

Modern quadrupeds

When discussing well-known robots, it is imperative to mention the work of Boston Dynamics, a renowned American company active in various areas of robotics. Despite being involved in multiple projects, some of their most notable contributions lie in the subfield of quadrupedal robots. However, the company's policies of not publishing papers about their work have resulted in a scarcity of literature on the subject, despite the widespread popularity of their creations. Their first major success came in 2008 with the creation of the second iteration of the Big Dog quadruped [27] (Figure 2.13). This robot, initially designed as a mule for the US military, demonstrated exceptional abilities in navigating uneven and challenging terrains, making it a pivotal landmark for future developments in the field. To address the noise

FIGURE 2.13: Big Dog by Boston Dynamics.

issues of Big Dog, Boston Dynamics introduced an improved version called

Legged Squad Support System (LS3) or AlphaDog in the following year. This version was more militarized and rugged to withstand harsh environments. Another version of the quadruped, Little Dog, was also released for technical research purposes, rather than military applications. In 2011, Boston Dynamics created another groundbreaking robot named Cheetah, which was designed to achieve high speeds in locomotion. By August 2012, Cheetah held the record for the fastest quadrupedal robot on a treadmill. One year later, Boston Dynamics released Wildcat, which became the fastest quadruped on rough terrain. Despite these impressive accomplishments, the company's most successful and popular creation is likely Spot (Figure 2.14). Spot was first introduced to the public on June 23rd, 2016, and since then it has garnered significant success in both the research field and in popular culture, with YouTube videos attracting millions of views. Spot also became one of the first sophisticated robots to be commercially available to the general public, owing to its design and ease of high-level control.

FIGURE 2.14: Spot by Boston Dynamics.

One institution that has made significant contributions to the development of modern quadruped robots is the Istituto Italiano di Tecnologia (IIT), with its HyQ robot [28] (Figure 2.15). Designed and developed by C. Semini and his team in 2010, the HyQ robot leverages the combination of hydraulic and electric actuation to achieve both high-speed and high-torque capabilities, while also maintaining compact overall dimensions. The design of the HyQ was further improved with the release of the HyQ2Max version in 2016 [29], which showed enhanced robustness and the added capability of self-righting.

In the same year that saw the development of HyQ in Italy, M. Hutter in Switzerland was engaged in the creation of StarlETH [30]. This robot was distinguished by its use of Series Elastic Actuators (SEA) in its actuation system, which enabled improved agility and efficiency as compared to Big Dog. The success of StarlETH prompted further research by ANYbotics, resulting

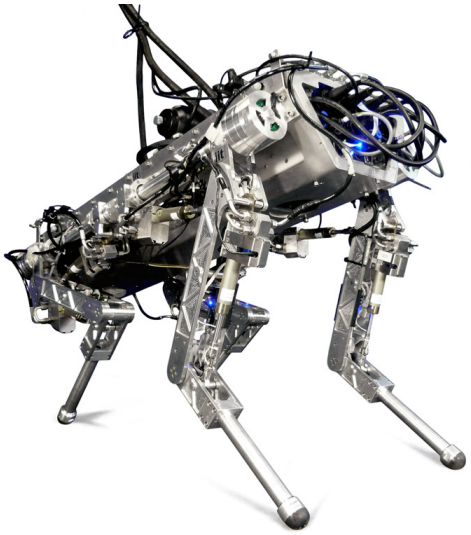

FIGURE 2.15: HyQ by IIT.

in the creation of ANYmal [31] (Figure 2.16), a more versatile and modular iteration of StarlETH. ANYmal, introduced in 2016, boasts exceptional robustness and joint mobility and has since been employed in a range of real-world applications.

FIGURE 2.16: ANYmal by ANYbotics.

At the Biomimetics Robotics Lab of the Massachusetts Institute of Technology (MIT) in Boston, a focus was placed on the development of a fast and energy-efficient quadruped robot, inspired by the work of Boston Dynamics' Cheetah. This led to the creation of the MIT Cheetah in 2013 [32]. The primary objective of its design was the reduction of energy losses, and this was achieved through a quadrupedal robot with a cost of transport of 0.5, which rivals that of running animals in nature and surpasses previous quadrupedal robots. The lab later introduced improved versions, the MIT Cheetah 2 [33] and MIT Cheetah 3 [34], which exhibited enhanced robustness and an

even lower cost of transport. Additionally, the lab developed Mini Cheetah [35] (Figure 2.17), a smaller and more economical version of their quadruped, serving as a research platform for the advancement of control systems for legged robots.

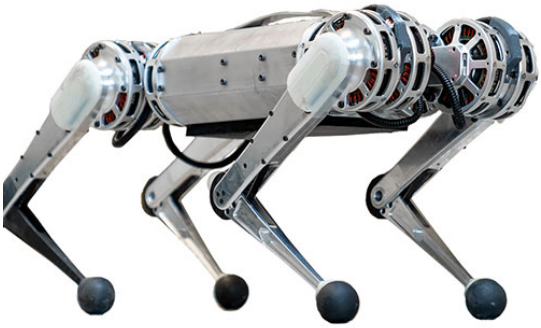

FIGURE 2.17: Mini Cheetah by MIT.

Finally, in 2018, the Chinese company Unitree introduced one of the most cutting-edge quadruped robots to date, known as the Unitree A1 (Figure 2.18). Although no information regarding the development of this robot is publicly available, it has demonstrated impressive capabilities, rivalling those of comparable robots, at a substantially reduced cost.

FIGURE 2.18: A1 by Unitree.

These are a few of the most renowned and significant quadruped robots developed to date, however, the field continues to progress, and it is anticipated that even more advanced models will be introduced in the upcoming years.

In the subsequent subsection, an overview of the general dynamic characteristics of quadruped robots will be presented.

2.2.2 Quadruped dynamics

To comprehend the dynamics of quadruped robots, it is essential to understand the meaning of a robot's dynamic model and its benefits compared to other models, such as kinematic models. The dynamic model of a robot, as stated by Featherstone [36], represents the relationship between the forces acting on the system and the accelerations they produce. This means that the dynamic model of a quadruped robot characterizes its motion and behaviour through its forces, moments, and accelerations, taking into account the effects of gravity, inertia, and actuator torques. This mathematical representation of the physical system is used to calculate the behaviour of the robot over time, given initial conditions and external inputs. In contrast, a kinematic model only defines the geometry and configuration of the robot and its joints, which provides limited information for control or simulation purposes. For these reasons, this paper will concentrate solely on the dynamic model of quadruped robots.

A pictorial representation of a planar dynamic model of a quadruped, as presented in [37], can be viewed in Figure 2.19.

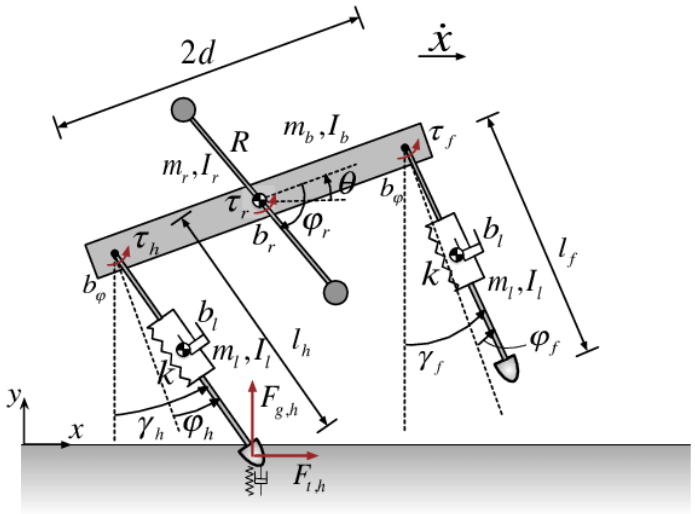

FIGURE 2.19: Representation of a planar dynamic model of a quadruped. Image from [37].

The study of dynamic models can encompass various types, including single-body and multi-body dynamics. Single-body dynamics focuses on the examination of the motion of a single, isolated body and the forces acting on it. Conversely, multi-body dynamics involves the analysis of the motion of multiple interconnected bodies and the interactions between these bodies in the system. Although the multi-body approach offers a more comprehensive and complex model, it necessitates a substantial amount of computational time for numerical analysis due to its large number of degrees of freedom.

Conversely, single-body dynamics models, despite being less accurate, are computationally simpler and enable real-time analysis [38]. Given the simplicity of single-body dynamics models, this review will concentrate mostly on this approach.

Another differentiation that can be established between dynamic models is the categorization into fixed-base and floating-base dynamics. The former assumes a rigid and stationary base to which the robot is attached, and the motion of the bodies is analyzed with reference to the base. Conversely, floating-base dynamics permit the base to exhibit motion and alter orientation during the system's motion. Consequently, when modelling a quadruped robot, the floating-base dynamics approach is considered the most appropriate.

Floating base dynamics

A significant distinction between fixed-base robots, such as manipulators, and floating-base robots is the level of actuation. Manipulators are typically considered to be fully actuated and unconstrained systems, while floating-base robots, such as quadrupeds, are underactuated. This is because the dynamics of a legged robot require the addition of an extra 6 degrees of freedom, representing the position and orientation of the robot with respect to an inertial frame, to the n degrees of freedom of the joints [39].

A simplified graphical representation of a quadruped considered as a floating-base system is depicted in Figure 2.20.

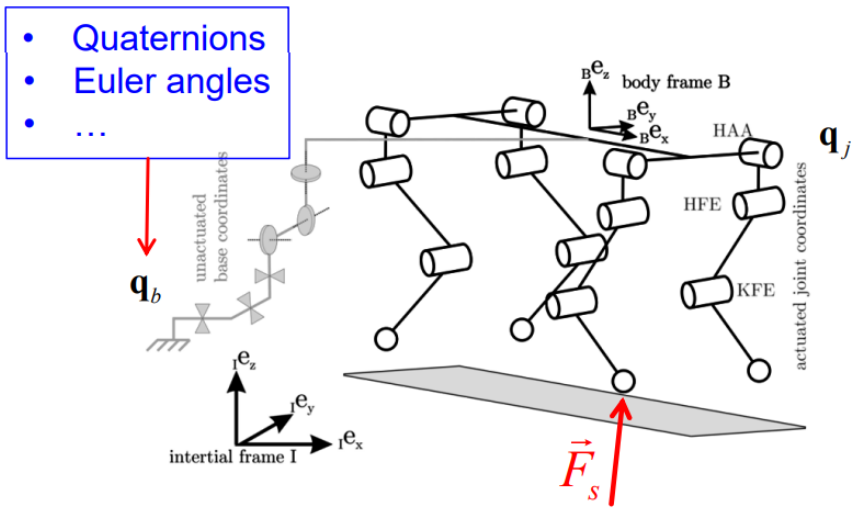

FIGURE 2.20: Representation of a quadruped as a floating-base system.

The mathematical representation of the system is outlined in [40] and will be briefly discussed in the following.

The system configuration is represented by the vector:

$$q = \begin{bmatrix} q_j & q_b \end{bmatrix}^T$$

where $q_j \in \mathbb{R}^n$ represents the actuated joint coordinate vector of the quadruped with n joints, and $q_b \in \mathbb{R}^6$ is the un-actuated coordinate vector of the reference system attached to the body with respect to an inertial frame. This vector includes the Cartesian position of the body, $(x, y, z) \in \mathbb{R}^3$, and the Euler angles describing its orientation, $(\phi, \theta, \psi) \in \mathrm{SO}^3$.

The equations of motion for a robot in contact with its environment can be represented as follows:

$$M(q)\ddot{q} + h(q, \dot{q}) = S^T\tau + J_C^T(q)\lambda$$

where:

- $M(q) \in \mathbb{R}^{n+6\times n+6}$ is the floating base mass matrix
- $h(q, \dot{q}) \in \mathbb{R}^{n+6}$ is the floating base centripetal, Coriolis and gravity forces
- $S = \begin{bmatrix} I_{n\times m} & 0_{n\times 6} \end{bmatrix}$ is the actuated joint selection matrix, ensuring that the applied torque only affects the actuated coordinates of the joints and not the un-actuated coordinates of the base
- $\tau \in \mathbb{R}^n$ is the vector of the applied torque
- $J_C \in \mathbb{R}^{k\times n+6}$ is the Jacobian of k linearly independent constraints
- $\lambda \in \mathbb{R}^k$ is the vector of k linearly independent constraint forces

It should be noted that the constraints are defined as points on the robot in contact with the environment where external forces or torques are applied and result in no motion with respect to the inertial frame.

However, due to the high dimensionality and complexity of the system, obtaining a fully precise model of the robot can be challenging and computationally intensive. As a result, reduced-order template models are often utilized in practical applications to address these computational difficulties and increase efficiency.

Reduced-order template models

The utilization of a full-order model for the control of quadruped robots presents significant challenges and difficulties. As a result, a majority of the studies in the field of quadruped robot control have employed reduced-order models, which provide a simplified representation of the system. There are various methods for obtaining a simplified model, but certain models have proven to be more effective and have gained widespread use in the academic community. These models are commonly referred to as "template models" [41] and will be the focus of the following discussion.

The single rigid body (SRB) model is widely used among template models for the control of quadruped robots. In this model, the body and legs of the robot are combined into a single rigid body (as depicted in Figure 2.21) and ground reaction forces (GRF) are utilized as inputs to control its position

and orientation. Despite its simplicity, the SRB model effectively captures the influence of the net external wrench on the centre of mass (CoM) motion and orientation, as explained by [42]. This model is preferred over full-order dynamic models when real-time constraints need to be met, and it is a good approximation of the robot model as the combined mass of the legs is typically less than 10% of the total mass, as indicated by [43].

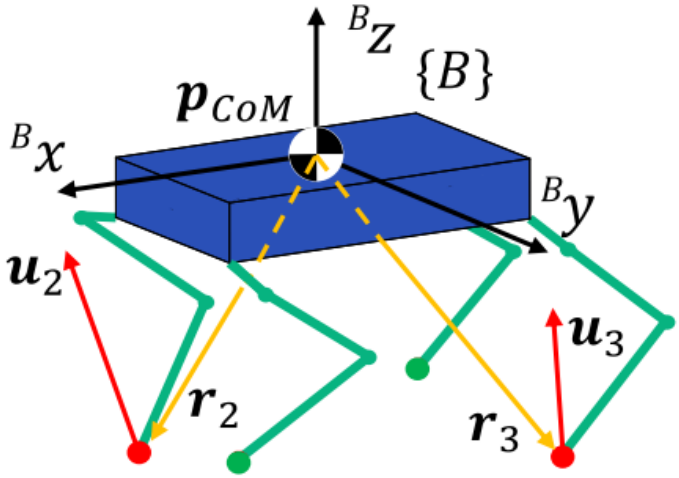

FIGURE 2.21: Representation of an SRB model of a quadruped. Image from [43].

A centroidal dynamics model (CDM) is a more complex reduced-order model that has proven to be highly efficient for biped robots, as demonstrated by Orin et al. in [44] (Figure 2.22). The CDM projects the full-body dynamics of an articulated model onto its centre of mass (COM), thereby simplifying the model and its motion to the centroidal inertia and the overall linear and angular momentum [45]. Furthermore, the model can be further reduced to a single rigid body model by using a frame anchored at the COM to represent the overall body orientation, as demonstrated in [46]. Unlike other models, the CDM takes into account the angular momentum in its dynamics, which makes it a more complex model [45]. However, the CDM remains more computationally efficient and easier to control than a full-order model, as it captures the essential dynamics of the robot without the need to model the detailed motion of its limbs.

The Linear Inverted Pendulum (LIP) model is another widely researched reduced-order model for quadrupeds [47], used mostly when studying static gaits. In this model, a legged robot is depicted as a point mass located on a massless leg (as illustrated in Figure 2.23). Despite not accurately capturing the full dynamics of the robot, the LIP model is frequently utilized due to the fact that many contemporary legged robots can be well-represented by this model, as they often have a heavily weighted upper body and relatively lightweight legs. The simplicity of the LIP model makes it analytically tractable, offering valuable physical insights into crucial robot behaviours.

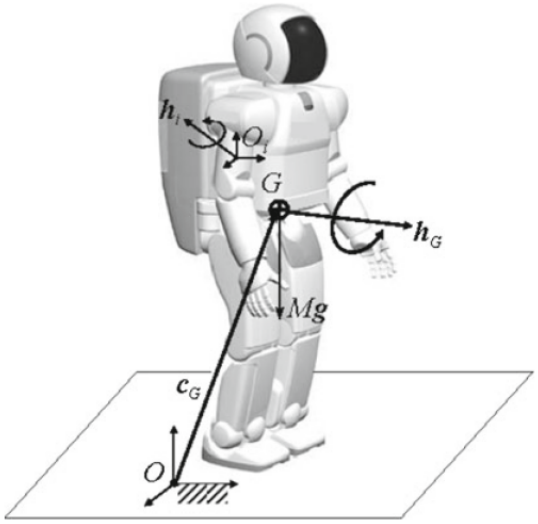

FIGURE 2.22: Representation of a CDM of a biped robot. Image from [44].

As a result, the LIP can be employed in motion planning to guarantee both computational efficiency and physical feasibility [48].

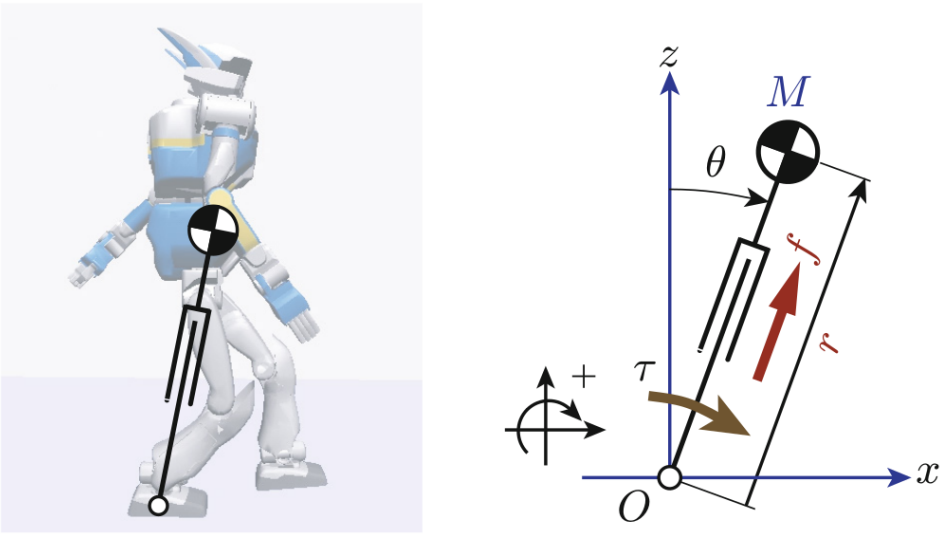

FIGURE 2.23: Representation of a LIP model of a biped robot. Image from [49].

While the previous model is suited for static gaits, the Spring Loaded Inverted Pendulum (SLIP) model is employed for the dynamic gait modelling of robots. This model embodies specific high-level control hypotheses regarding the coordination of joints and limbs by animals or robots, as described in [50]. The quadruped is depicted as a single rigid body that is sustained by a leg that is massless and comprised of a spring-damper system (as illustrated in Figure 2.24). The SLIP model accurately captures the centre of mass (CoM) dynamics of humans and animals while they are hopping and running in the sagittal plane [51]. By utilizing a SLIP model, engineers

can analyze the motion of quadrupeds and comprehend the interaction between its leg dynamics and body dynamics, without having to account for the complexities associated with the full multi-body system.

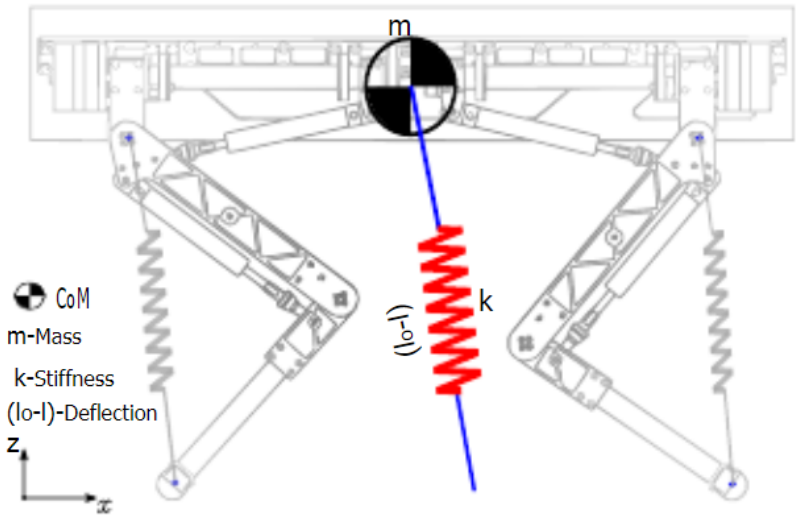

FIGURE 2.24: Representation of a SLIP model of a quadruped robot. Image from [52].

2.3 Chapter summary and conclusion

This chapter serves as a preliminary discussion of quadruped robots and legged locomotion.

In Section 2.1, an overview of legged locomotion is presented, highlighting its key characteristics such as the ability to use discrete footholds when moving and allowing dynamic stability of the system. Additionally, Subsection 2.1.1 provides a comparison between legged and wheeled locomotion, emphasizing the differences in complexity, energy efficiency, and adaptability to rough terrains.

In Section 2.2, the focus is specifically on quadruped robots. Subsection 2.2.1 covers the historical development of quadrupeds, citing significant milestones in the field. This is followed by a discussion of the most notable quadruped robots in use today. Subsection 2.2.2 addresses the dynamical aspect of quadrupeds, beginning with a general description of floating base robots, and then briefly introduces the four most popular types of reduced-order template models used for quadrupeds: SRB, CDM, LIP, and SLIP.

In summary, this chapter provides a brief introduction to the main subject of this review: quadruped robots. The next chapter will delve into the techniques employed for learning models of complex systems.

Chapter 3

Techniques for learning the dynamics of complex systems

The dynamics of complex systems can be learned through different techniques, including system identification using experiments and simulations, or time series analysis through the examination of patterns and trends in collected data. In this chapter, the focus will be on machine learning techniques for learning the dynamics of complex systems. An overview will be provided of various techniques, ranging from traditional methods to cutting-edge techniques based on deep learning and genetic programming.

The application of these machine learning techniques to learn the dynamics of complex systems is called model learning. It can be defined as the process of automatically discovering the mathematical or statistical structure and parameters of a model from a set of data or observations. The objective is to find a model that accurately represents the relationship between inputs and outputs in the data and generalizes effectively to new, unobserved data.

The following sections will outline some of the most important model learning techniques.

3.1 Classic techniques

The current section will present a brief overview of some of the most commonly utilized techniques for model learning that do not necessitate the utilization of complex tools, like neural networks or evolutionary algorithms, and their implementation in the case of non-linear dynamical systems. These techniques are referred to as "classic techniques" due to their relative simplicity.

The methods discussed below are categorized as forms of regression, a statistical technique utilized to establish the association between a dependent variable and one or more independent variables [53]. The primary objective of regression analysis is to determine a relationship that closely matches the data points in order to reduce the discrepancy between the actual and predicted values of the dependent variable. Subsequently, the regression model can be employed to make predictions regarding the dependent variable with reference to novel values of the independent variables.

3.1.1 Linear regression

In particular, linear regression is a specific type of regression analysis that models the relationship between a dependent variable and one or more independent variables using a linear equation. There are two main types of linear regression: simple linear regression and multivariate linear regression.

The first case is utilized to determine the relationship between a single dependent variable and a single independent variable. This method involves fitting a linear equation to the data that optimally represents the relationship between the two variables.

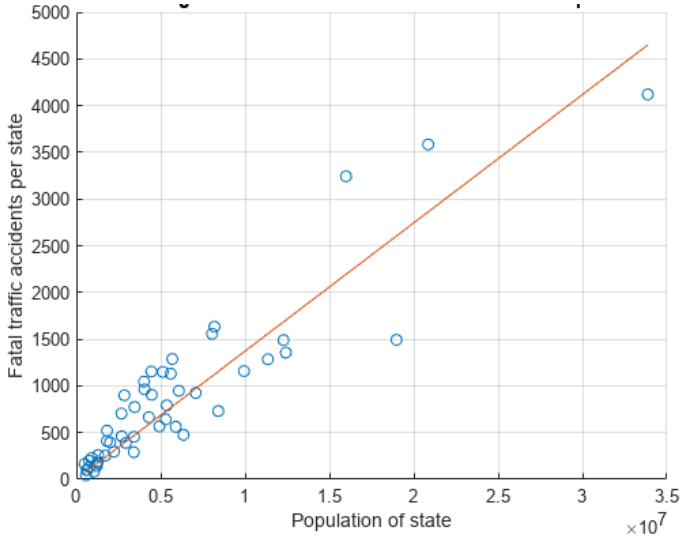

FIGURE 3.1: Example of simple linear regression. Image from MathWorks.

A simple linear regression model is represented by:

$$y = \beta_0 + \beta_1 x + \epsilon$$

where ϵ is the error term, β_0 is the intercept and β_1 is the slope of the regression line.

On the other hand, multivariate linear regression studies the relationship between a dependent variable and multiple independent variables. This method employs the fitting of a hyperplane to the data in order to depict the relationship between the dependent variable and the multiple independent variables in the most accurate manner possible (Figure 3.2).

A multivariate linear regression model with k predictor variables is represented by:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_k x_k + \epsilon$$

where ϵ is the error term, β_0 is the intercept and $\beta_1, \beta_2, ..., \beta_k$ are the partial regression coefficients [54].

The equation can be rewritten in matrix form as:

$$y = X\beta + \epsilon$$

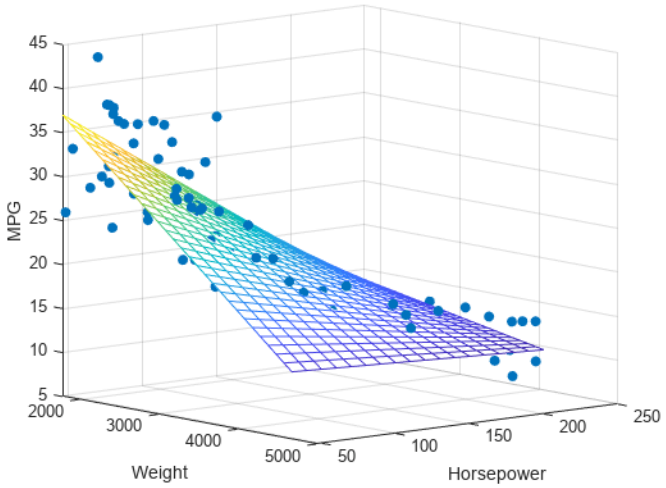

FIGURE 3.2: Example of multivariate linear regression. Image from MathWorks.

and thus the regression model β is estimated using the least square estimates as:

$$\hat{\beta} = (X^T X)^{-1} X^T y$$

It is evident that the two methods only differ from one another in terms of the number of independent variables utilized for modelling the relationship with the dependent variable.

Application in autoregressive models

Linear regression is based on the assumption of a linear relationship between the independent and dependent variables, and thus may not be an appropriate approach for modelling non-linear dynamical systems such as quadrupeds. Nonetheless, several techniques have been proposed to enable the use of linear regression for modelling such systems.

One intuitive solution to deal with the non-linearity involves approximating a small region of the non-linear system with a linear system and then applying linear regression to estimate the parameters of the linear system within that region. This process can be repeated for multiple regions to create a piecewise linear model of the non-linear system.

On the other hand, to tackle the dynamic aspect of the system, a valid approach is the use of Autoregressive (AR) models, which are a type of time-series model that leverages past observations of a variable to forecast future values of that variable. One widely used type of AR model is the AutoRegressive model with eXogenous inputs (ARX) [55], or its extension, the AutoRegressive Moving Average model with eXogenous inputs (ARMAX) [56].

ARX is a linear regression model that uses past values of the time series and exogenous variables (i.e., inputs from outside the system) to make predictions about the future values of the time series. The model can be expressed mathematically as [57]:

$$y = \beta_0 + \sum_{i=1}^{p} \beta_i y_{t-i} + \sum_{i=1}^{q} \gamma_i u_{t-i} + \epsilon_t$$

In this equation, y_t represents the output variable at time t, u_t represents the input or exogenous variable at time t, p denotes the order of the autoregressive component which represents the number of past values of the output variable used in the model, q denotes the order of the exogenous input component which represents the number of past values of the input variable used in the model, $\beta_0, ..., \beta_p$ are the coefficients for the autoregressive component which determine the impact of past values of the output variable on the current value, $\gamma_1, ..., \gamma_q$ are the coefficients for the exogenous input component which determine the impact of past values of the input variable on the current value, and ϵ_t is the error term at time t. The coefficients $\beta_0, ..., \beta_p$ and $\gamma_1, ..., \gamma_q$ are estimated using linear regression to obtain the final ARX model. In this way, the linear regression model can capture any linear relationships between the inputs and outputs, while the ARX model can capture any time-varying dynamics and non-linearities.

ARMAX is an extension of ARX that includes a moving average component to model the impact of past errors on the current output. This model can be preferred over ARX when significant exogenous factors that influence the system being modelled are present, such as external disturbances like the wind.

3.1.2 Support vector regression

Support Vector Regression (SVR) is a more sophisticated form of regression that involves more complex calculations. SVR utilizes Support Vector Machines (SVMs) to model non-linear relationships between the input features and output variables. The fundamental concept behind SVR is to obtain a hyperplane that optimally fits the data in a high-dimensional feature space by minimizing the prediction error. In the case of SVR, the hyperplane takes the form of a regression line or curve that passes as close as possible to the maximum number of data points while ensuring that the prediction error remains within a specific margin or threshold (Figure 3.3).

In the context of regression analysis, the mathematical representation of the regression line is typically denoted as $f(x)$. This line is defined by the equation:

$$f(x) = w \cdot x + b$$

where w represents the gradient of the function and b is a constant term.

The goal of Support Vector Regression (SVR) is to create a hyperplane that is as flat as possible, ultimately minimizing the norm of the weight vector, w.

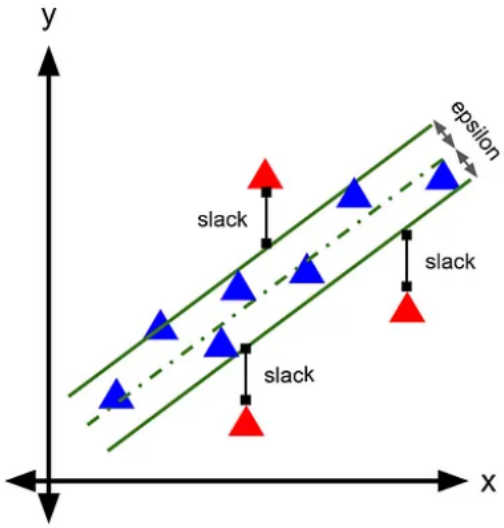

FIGURE 3.3: Example of Linear Support Vector Regression. Image from [58].

This approach can be expressed as a convex optimization problem [59]:

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{l}(\xi_i + \xi_i^*) \\ \text{subject to} \quad & \begin{cases} y_i - w \cdot x_i - b \leq \varepsilon + \xi_i \\ w \cdot x_i + b - y_i \leq \varepsilon + \xi_i^* \\ \xi_i, \xi_i^* \geq 0 \end{cases} \end{aligned}$$

In this formulation, ε represents the maximum deviation allowed from the desired targets, y_i, and slack variables ξ_i and ξ_i^* are introduced to relax hard constraints that cannot be satisfied with a linear hyperplane. The constant $C > 0$ controls the trade-off between the flatness of the hyperplane and the tolerance of deviations larger than ε.

Support Vector Regression can be classified as either linear or non-linear, depending on the type of kernel function applied to transform input features into a higher-dimensional space [60]. In contrast to the linear SVR, the non-linear version uses non-linear kernels, such as the radial basis function, polynomial or sigmoid kernels, to capture complex and non-linear relationships between the input features and the output variables, as shown in Figure 3.4.

Application to non-linear dynamical systems

In the context of modelling non-linear dynamical systems, Support Vector Regression (SVR) has demonstrated notable advantages over linear regression. By leveraging a kernel function to transform input data into a higher-dimensional space, SVR is able to model non-linear relationships that may exist between input and output variables. This enables the input data to be mapped from the original feature space to a new feature space where it may be more easily separable.

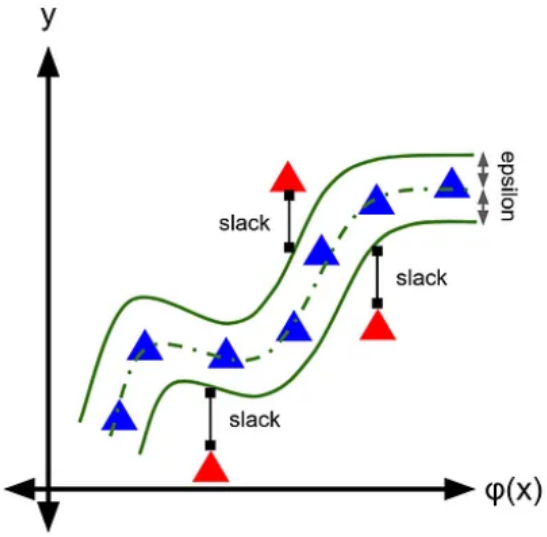

FIGURE 3.4: Example of Non-Linear Support Vector Regression. Image from [58].

However, when working with dynamical systems, it is often necessary to treat input and output variables as time series data. In this case, an ARX model, as described in 3.1.1, can once again be implemented by using past values of input and output variables as features in the SVR model [61]. The ARX model captures the system dynamics, while the SVR model models non-linear relationships between input and output variables.

The actual mathematical procedure for implementing this approach is rather extensive and is not described in this section. The general procedure involves generating a feature matrix using past values of input and output variables and then applying SVR to learn a non-linear function that maps the feature matrix to the output variable. The ARX model is utilized to generate feature vectors for new input data.

3.1.3 Decision tree regression

Decision tree regression is a non-parametric supervised learning technique that models the relationship between input features and output values by constructing a tree-like structure. This type of regression analysis aims to predict a continuous numerical output value for a given input vector of features, and it accomplishes this by recursively partitioning a dataset into smaller subsets while simultaneously constructing a decision tree [62]. The decision tree comprises decision nodes, representing rules, and leaf nodes, representing predictions of the output value (see Figure 3.5). After the decision tree is trained, it can be used to predict the output value for new input vectors by following the decision rules in the tree from the root to a leaf node and outputting the prediction at that leaf node.

Ensemble methods such as random forest regression can combine decision trees to improve their performance and reduce overfitting. Random forest regression is an additive model that makes predictions by combining

FIGURE 3.5: Example of decision tree regression.

decisions from a sequence of base models, resulting in a cumulative output of decision trees (see Figure 3.6) [63].

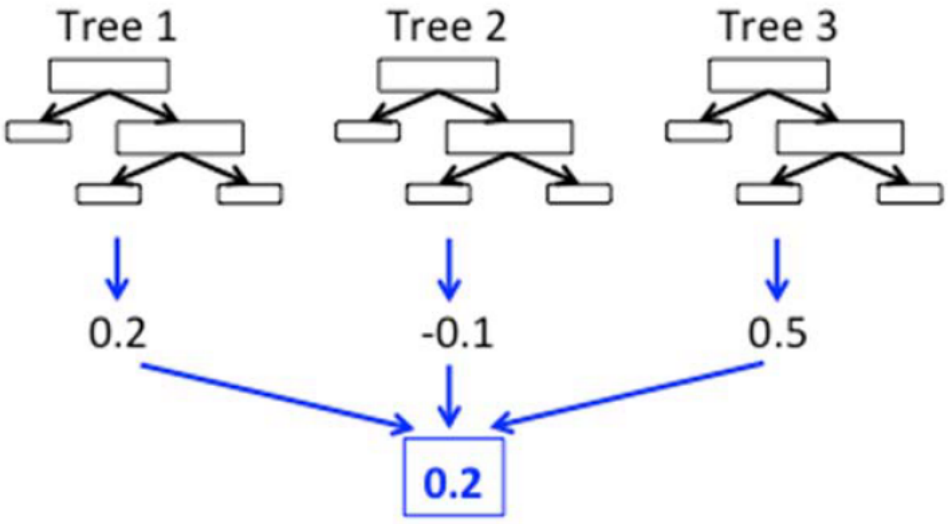

FIGURE 3.6: Example of random forest regression.

In the context of non-linear systems, decision tree regression is another effective technique alongside SVR. One of its advantages over SVR is that it produces more interpretable and transparent models, as the tree structure allows for a clear understanding of the decision-making process and the relative importance of different input variables in predicting the output. Furthermore, time-series data can be used as input to decision trees, making them suitable for dynamic systems.

3.1.4 Gaussian regression

Gaussian regression is a probabilistic regression analysis method that employs Gaussian processes (GPs) to model the association between input and output variables. A GP is "a collection of random variables, any finite number of which have (consistent) joint Gaussian distributions" [64]. In the context of regression analysis, a GP can be utilized to model the distribution of

plausible functions that could link the input and output variables. This enables the flexible and non-parametric modelling of complex relations, without making any assumptions about the functional relationship.

To perform Gaussian regression, a GP is first characterized by a mean function and a covariance function, also known as a kernel function, that define the prior distribution over functions. The GP is expressed as:

$$f(x) \sim \mathcal{GP}(m(x), k(x, x'))$$

where $f(x)$ represents the function value at input x, $m(x)$ is the mean function that determines the expected output value given the input, $k(x, x')$ is the covariance function that specifies the correlation between the function values at input x and input x', and the symbol $\sim$ denotes that $f(x)$ is a random variable drawn from the GP prior distribution. The kernel function dictates the shape of the prior distribution and can be customized to the specific problem.

Once the GP prior is established, the model is trained using a set of input-output data pairs. Based on this data, the GP posterior distribution is computed, which provides the most probable function(s) that could have generated the observed data (Figure 3.7). The posterior distribution delivers a probabilistic estimation of the output for any given input value, as well as an estimation of the uncertainty associated with this prediction.

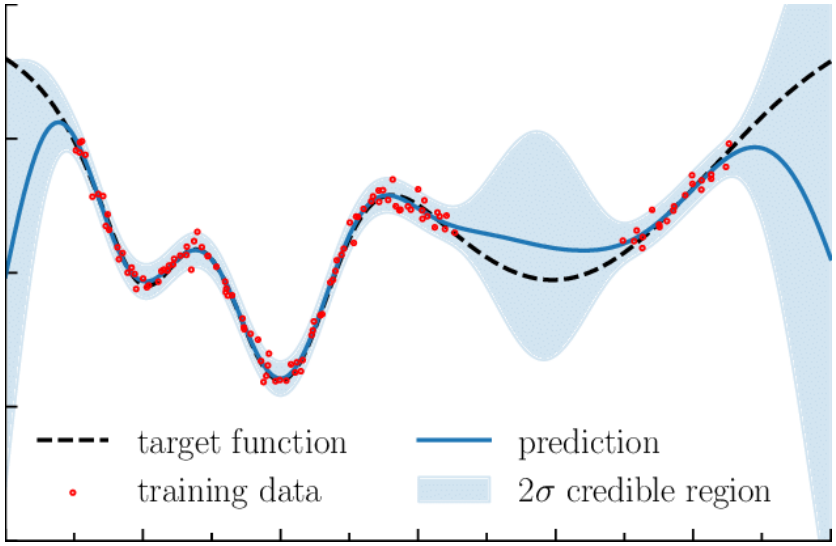

FIGURE 3.7: Example of Gaussian regression. Image from [65].

In general, Gaussian regression presents numerous advantages over other regression techniques, such as the capacity to model complex and non-linear associations, provide a probabilistic estimation of the output and its uncertainty, and ease of use and interpretability. Consequently, it is currently among the most commonly used classic techniques for learning complex models [66].

Its ability in modelling non-linear associations between input and output variables, while also providing a probabilistic evaluation of the output and its corresponding uncertainty, is particularly significant when considering unstable behaviour that may result from uncertainty in the system's state

estimation. Thus, the application of Gaussian regression can result in an enhancement of the system's stability and control.

When dealing with dynamic systems, the primary focus is on modelling the temporal evolution of a system, which can be described through a set of differential equations. In such cases, the input to the model includes not only a static feature vector or time series but also the previous system state. Therefore, Gaussian processes can be utilized to model the distribution of feasible functions that represent the system's evolution over time, as shown in previous research [67].

3.2 Deep learning

A more computationally intensive, yet robust method of understanding the dynamics of complex systems involves the application of deep learning. Deep learning is a representation-learning approach that utilizes multiple levels of representation, achieved by combining simple, non-linear modules that modify the representation at one level (initially the raw input) into a representation at a higher level that is marginally more abstract [68].

These modules are called artificial neural networks (ANNs), which are computational models inspired by the structure and function of biological neural networks. ANNs are composed of interconnected neurons arranged in layers, including an input layer, one or more hidden layers, and an output layer (Figure 3.8). During training, the weights between neurons are adjusted to minimize the discrepancy between actual and expected output, using a method called backpropagation. Activation functions introduce non-linearities into the network, allowing it to learn complex relationships between input and output.

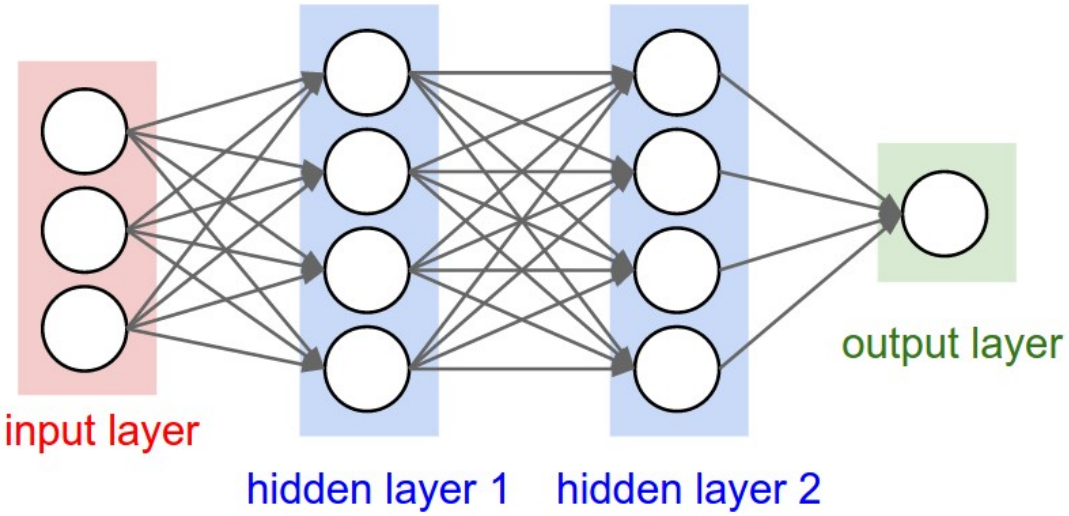

FIGURE 3.8: Pictorial representation of an artificial neural network. Image from [69].

Certain variations of these neural networks are particularly well-suited for modelling the dynamics of complex systems. In particular, recurrent neural networks (RNNs) are suitable for learning time-series data, transformers are an innovative powerful method widely used in natural language processing, and deep autoencoders are used to learn a condensed representation of the input data. These techniques will be briefly outlined in the following subsections.

3.2.1 Recurrent Neural Networks

A recurrent neural network (RNN) is a type of neural network architecture that is designed to work with sequential data by maintaining an internal state or "memory" that captures information about the previous inputs seen so far. RNNs achieve this by using a feedback loop that allows information to be passed from one step of the sequence to the next [70] (Figure 3.9).

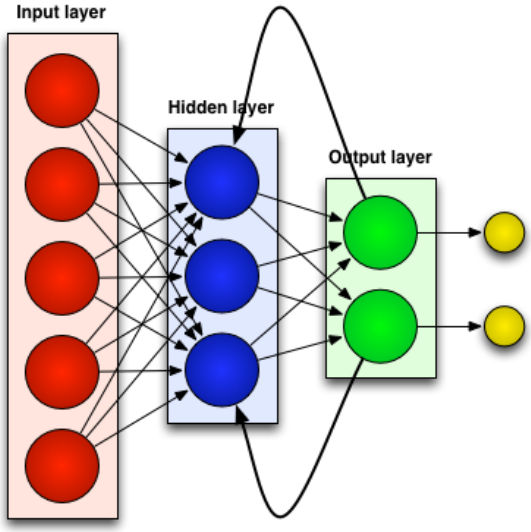

FIGURE 3.9: Pictorial representation of a recurrent neural network. Image from [71].

At each time step t, the RNN takes an input vector x_t and an internal state vector h_{t-1} from the previous time step as input. The updated internal state h_t and the output y_t are then computed using the following equations:

$$h_t = f(W_{hh}h_{t-1} + W_{hx}x_t + b_h)$$
$$y_t = g(W_{yh}h_t + b_y)$$

In these formulas, W_{hh} and W_{hx} are weight matrices that determine how much the previous internal state and current input contribute to the updated state at each time step, respectively. W_{yh} is a weight matrix that maps the internal state to the output. b_h and b_y are bias terms. $f()$ and $g()$ are activation functions, which may be different depending on the specific architecture of the RNN.

Thus, assume the presence of a non-linear dynamical system of the form:

$$x_{t+1} = f(x_t, u_t)$$

where x_t is the state of the system at time t, u_t is the input at time t, and $f()$ is a non-linear function that describes how the system evolves over time. The RNN can be used to learn the dynamics of this system by feeding the current state x_t and input u_t into the network, and training the network to predict the next state x_{t+1}.

The ability to deal with sequential data has made these networks very powerful tools to model dynamical systems and there are many examples of it in literature, such as [72].

However, vanilla RNNs suffer from the "vanishing gradient" problem, which can make it difficult to learn long-term dependencies in the input sequence. To address this issue, more advanced types of RNNs have been developed, such as long short-term memory (LSTM) networks and gated recurrent units (GRUs), which use additional mechanisms to control the flow of information through the network and alleviate the vanishing gradient problem.

Long Short-Term Memory networks

As previously elucidated, traditional RNNs may struggle to learn long-term dependencies in input sequences due to the problem of vanishing gradients. Specifically, the gradient of the error function with respect to the network's parameters can become very small, impeding the ability to update the weights during training. This issue can be especially pronounced in deep RNNs, where the gradient may diminish exponentially as it propagates back through time.

In contrast, Long Short-Term Memory (LSTM) networks [73] utilize a more sophisticated architecture that incorporates "memory cells" and "gates" to regulate information flow through the network. Memory cells are designed to store information for prolonged periods, while gates govern the input, output, and erasure of information from memory cells. This enables LSTM networks to selectively remember or forget information over extended periods, thereby avoiding the vanishing gradient problem.

As illustrated in Figure 3.10, an LSTM memory cell consists of a cell state that stores information over time, an input gate that governs information inflow into the memory cell, a forget gate that determines which information to retain and which to discard from the cell's memory state and an output gate that determines what information to output from the cell's memory state.

LSTM networks are purposefully designed to selectively store and retrieve information over time, a crucial feature in modelling dynamic systems that evolve over time. As evident in [75], LSTM networks are particularly suited to modelling these types of systems since they can capture long-term dependencies in sequential data. Additionally, the capacity of LSTMs to filter out irrelevant information and noise by selectively storing and retrieving

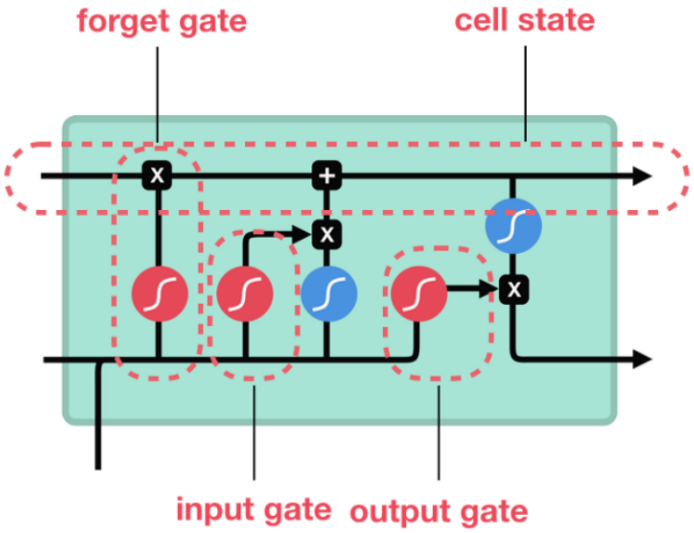

FIGURE 3.10: Pictorial representation of an LSTM memory cell. Image from [74].

information over time is fundamental in modelling complex systems with numerous interacting factors.

Gated Recurrent Units

A more recent solution to address the vanishing gradient problem was proposed in 2014, in the form of Gated Recurrent Units (GRUs) [76]. Compared to LSTMs, GRUs achieve a similar outcome with a simpler and more computationally efficient structure that comprises only two gates: an update gate and a reset gate (see Figure 3.11). The update gate governs how much of the prior hidden state to retain and how much of the new input to include in the hidden state. Meanwhile, the reset gate regulates how much of the prior hidden state to exclude while computing the new hidden state.

In general, both LSTMs and GRUs are appropriate for modelling non-linear dynamical systems. LSTMs are often regarded as more expressive and capable of capturing more complex relationships in sequential data, which makes them well-suited for tasks that require precise modelling of long-term dependencies and fine-grained control over information flow. However, GRUs are less complex and computationally efficient than LSTMs, which makes them suitable for tasks that demand faster training times and larger datasets, such as real-time applications [77].

Comparison with NARX networks

A different approach to utilizing neural networks for learning dynamical systems from time-series data is through the use of non-linear autoregressive networks, specifically the Nonlinear Autoregressive with eXogenous inputs (NARX).

Unlike the ARX network discussed in Subsection 3.1.1, the NARX network is better suited for learning non-linear models. While ARX models

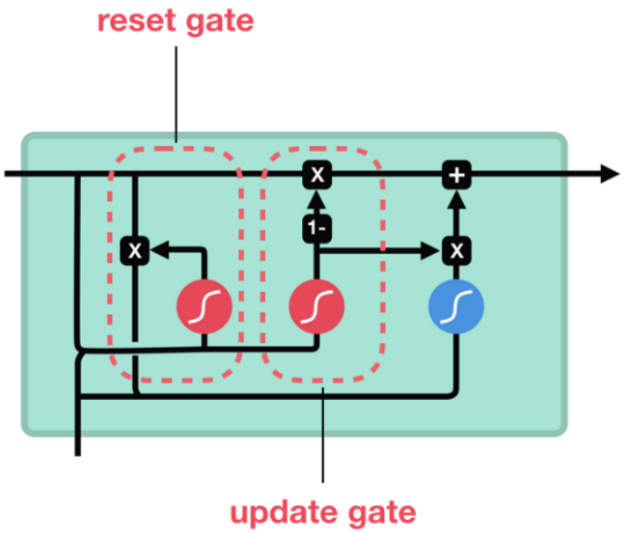

FIGURE 3.11: Pictorial representation of a GRU memory cell. Image from [74].

are commonly paired with linear regression or SVR to learn linear systems, NARX networks typically employ feedforward neural networks to capture more complex non-linear dynamics. Additionally, while ARX networks include only past input and output values as inputs, NARX networks also include previous predictions, which further enhances their ability to model non-linear systems [78].

NARX networks possess both strengths and weaknesses in comparison to recurrent neural networks. NARX models are feedforward neural networks with feedback loops, which makes them better suited for modelling systems with external inputs. On the other hand, RNNs are neural networks with recurrent connections that allow for the passing of information between time steps, making them better suited for modelling sequences of data with complex temporal patterns and long-term dependencies.

To merge the strengths of both techniques, recurrent NARX neural networks have been proposed and have demonstrated impressive performance and efficiency in various applications [79].

3.2.2 Transformers

Transformers, an innovative neural network model, were first introduced in 2017 in the work of Vaswani et al. [80]. They have been shown to be the current best solution for natural language processing tasks, surpassing previous models such as Recurrent Neural Networks (RNNs) and Convolutional Neural Networks (CNNs).

The architecture of transformers, illustrated in Figure 3.12, is complex and a detailed description of it is beyond the scope of this review. However, the main features of this technique will be briefly discussed.

The key innovation of the transformer model is the self-attention mechanism, which enables parallel processing of input sequences, as opposed to

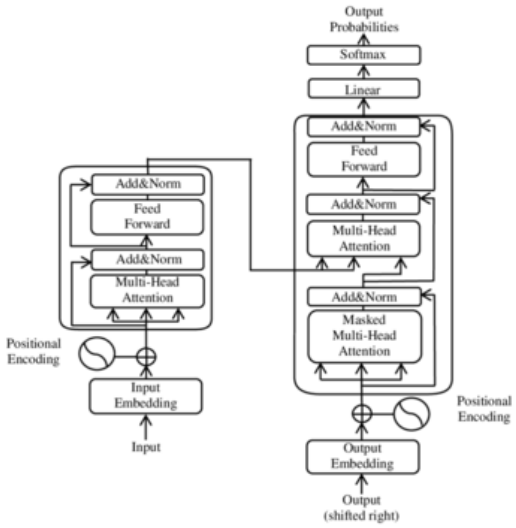

FIGURE 3.12: Model architecture of a transformer. Image from Wikipedia.

the sequential processing performed by RNN models. The transformer model also includes multi-head attention, feedforward networks, and layer normalization, which all contribute to its exceptional performance. The multi-head self-attention mechanism allows the model to attend to different parts of the input sequence, while the position-wise feedforward network applies linear and non-linear transformations to each position in the sequence independently. This feature enables the system to capture long-range dependencies more effectively than RNN-based models.

Application to physical systems

As aforementioned, transformers have shown remarkable efficiency in the field of natural language processing (NLP), and most of the literature regarding this technique is tailored to that domain. However, attempts have been made to extend their application to the modelling of physical systems.

For instance, [81] propose a novel approach to leverage transformers to model dynamical systems. Specifically, their method employs Koopman-based embeddings to map any dynamical system into a vector representation, which can be predicted by a transformer. The authors demonstrate that their proposed model outperforms traditional surrogate models in accurately predicting various dynamical systems, highlighting the potential of transformers to offer advantages over conventional surrogate methods for physical system modelling.

Despite promising results, scaling such deep learning models and developing generalizable time-cognizant capabilities to predict multi-time-scale phenomena present in physical systems requires further investigation. Thus, more research is needed to fully realize the scope of transformers' potential in the context of physical systems.

3.2.3 Autoencoders

While the previous sections discussed techniques for learning the model of a complex system using input-output pairs, this subsection focuses on a topic that can complement model learning approaches. Specifically, it explores the use of autoencoders to reduce the dimensionality of input data [82].

Autoencoders are a self-supervised type of artificial neural network. The term self-supervised learning connotes that the feedback to the prediction is furnished by the initial input itself, making the algorithm self-sufficient without any human contribution. Essentially, autoencoders learn a compressed representation or encoding of the input data and subsequently use this encoding to reconstruct the original input data as closely as possible.

By using the mathematical notation of [83], an autoencoder is a type of feedforward neural network designed to learn the identity mapping $h : x \mapsto \tilde{x}$ with $x \approx \tilde{x}$ and $h : \mathbb{R}^{n_x} \to \mathbb{R}^{n_x}$. Its first primary component is the encoder, defined as $h_{enc} : x \mapsto \hat{x}$ with $h_{enc} : \mathbb{R}^{n_x} \to \mathbb{R}^{n_{\hat{x}}}$ and $n_{\hat{x}} \ll n_x$, which maps a high-dimensional vector x to a lower-dimensional latent space representation $\hat{x}$. The second component, the decoder, is $h_{dec} : \hat{x} \mapsto \tilde{x}$ with $h_{dec} : \mathbb{R}^{n_{\hat{x}}} \to \mathbb{R}^{n_x}$ and maps the code $\hat{x}$ to an approximation of the original high-dimensional vector $\tilde{x}$ (Figure 3.13).

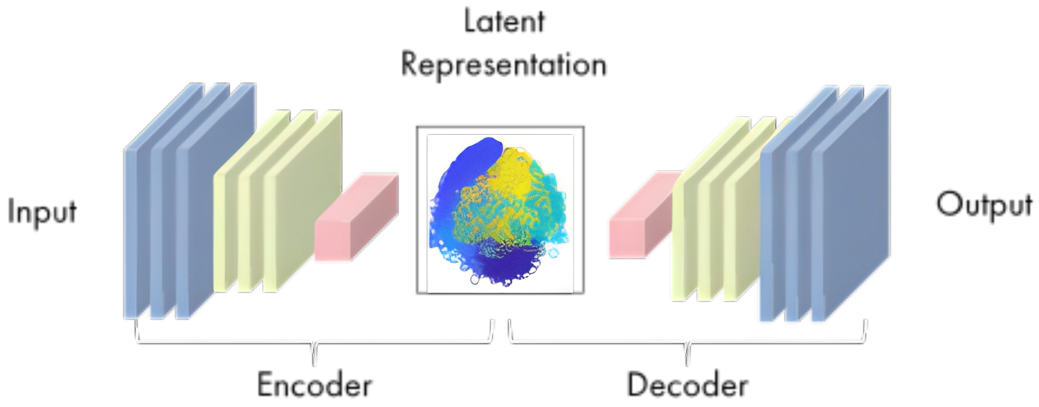

FIGURE 3.13: Pictorial representation of an autoencoder. Image from MathWorks.

Thus, the resulting autoencoder takes the form:

$$h : x \mapsto h_{\text{dec}} \circ h_{\text{enc}}(x)$$

The autoencoder is optimized to minimize the reconstruction error between the input data x and the output data $\tilde{x}$. This encourages the autoencoder to learn a compressed representation that captures the most important features of the input data, such that if $h(x) \approx x$ over a dataset $x \in \{x^{(i)}\}_i$, then the low-dimensional codes $h_{\text{enc}}(x^{(i)})$ contain sufficient information to recover accurate approximations of the data $\hat{x}^{(i)} \approx x^{(i)}$ via the application of the decoder h_{dec}.

Convolutional autoencoders

A variant of the vanilla autoencoder that has gained attention is the convolutional autoencoder, which incorporates the principles of convolutional neural networks (CNNs) (Figure 3.14). Convolutional autoencoders are particularly suitable for image data and other data types with a spatial structure because they leverage the local correlation present in the data. This approach can be more effective than vanilla autoencoders in learning the dynamic model of a non-linear dynamical system where the input data has a spatial structure, such as videos or neuroscience systems. A demonstration of this approach is exemplified in [83].

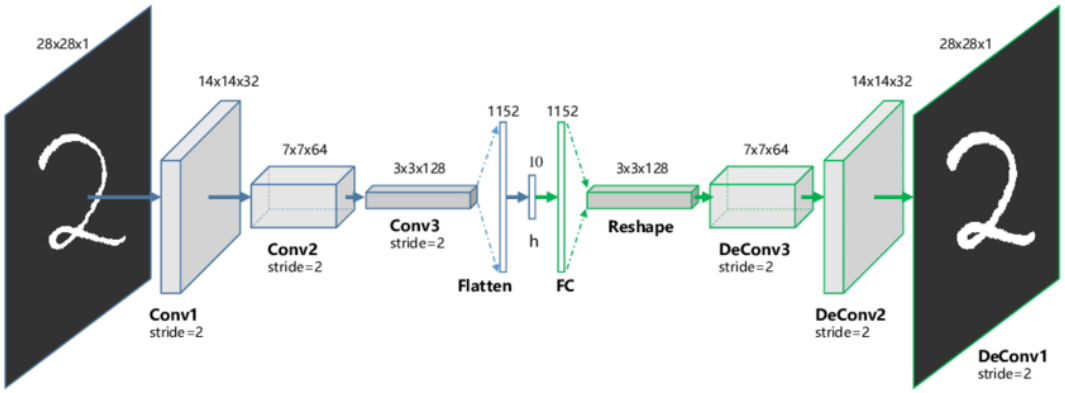

FIGURE 3.14: Pictorial representation of a convolutional autoencoder. Image from [84].

Furthermore, convolutional autoencoders can capture hierarchical features in the data by utilizing multiple convolutional layers with varying filter sizes. This characteristic can make them more effective than vanilla autoencoders in learning the dynamic model of a non-linear dynamical system where the input data has a hierarchical structure.

Convolutional autoencoders can also be stacked to learn and encode different features in the data, such as both spatial and temporal dimensions, as exemplified in [85].

Role in learning dynamics of complex systems

As mentioned earlier, unlike RNNs, autoencoders are not primarily used for learning a model using input-output pairs. Rather, they are frequently used in conjunction with other techniques to reduce the dimensionality of the input data by extracting and retaining only the most important features, thus enhancing the performance, feasibility, and computational efficiency of the learning algorithm.

However, autoencoders can still play a role in learning complex models by being combined with other techniques. The basic idea behind this approach is to first train an autoencoder to encode the input data in a compressed, lower-dimensional form. This encoded data is then fed into other techniques that learn the input-output relationship. For instance, in a study

presented in [86], autoencoders were combined with various regression techniques, with Support Vector Regression, outlined in Subsection 3.1.2, producing the best results. Another study presented in [87] combined autoencoders with another deep learning technique, specifically Long Short-Term Memory (LSTM) networks discussed in Subsection 3.2.1.

Autoencoders can also be trained in conjunction with latent dynamics, as demonstrated in [88]. The authors of this paper introduce deep learning autoencoders to discover coordinate transformations that capture the underlying parametric dependence of a dynamical system, expressed in terms of its canonical normal form. To achieve this, a mixed loss function is utilized: it combines the reconstruction error from the autoencoders with a loss term that encourages the latent dynamics to conform to a given normal form. By training the autoencoders in this manner, the appropriate coordinate transformation can be learned, and the underlying dynamics of the system can be accurately captured.

3.3 Symbolic regression and Equation discovery

Equation discovery techniques refer to a type of machine learning method that aims to identify mathematical equations or models that best describe the relationship between input and output variables. Applied to complex processes, they can provide insight into the relationships between different variables and improve predictions or control of the system [89].

There are various techniques used for equation discovery, a few of which will be described below.

3.3.1 Genetic programming

Genetic programming, first introduced in [90], is a branch of machine learning and evolutionary computation (Figure 3.15) that utilizes the principles of natural selection and genetics to create computer programs capable of solving a specific problem. It involves the use of genetic operators, such as mutation, crossover, and selection, to produce a population of programs that are evaluated for their fitness based on their ability to solve the problem.

The programs are treated as genomes that can be evolved and manipulated using genetic operations. The process begins with a population of randomly generated programs that are evaluated based on their fitness, and the fittest individuals are selected to reproduce. The genetic material of these individuals is then passed on to the next generation, while weaker ones are eliminated from the population. Over time, the population evolves, with the fittest individuals surviving and reproducing, resulting in programs that can effectively solve the problem.

Gene expression programming

Since the inception of genetic programming, numerous advancements have been made to enhance the technique by incorporating additional features

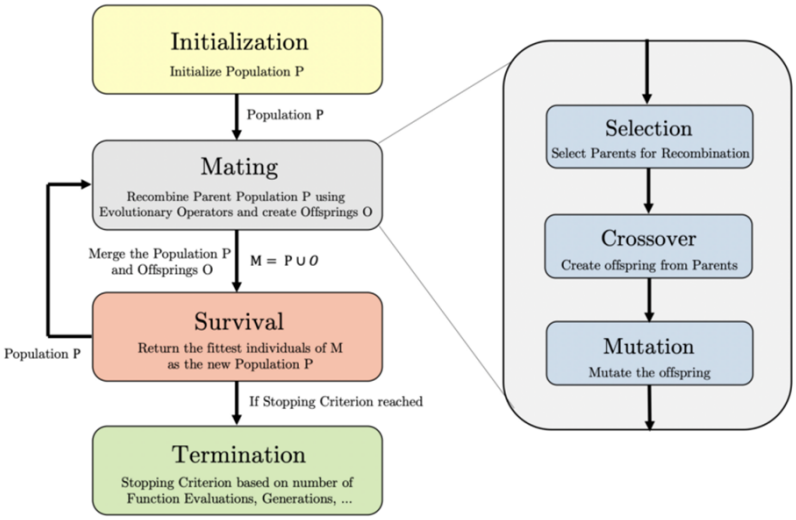

FIGURE 3.15: Working scheme of an evolutionary algorithm.

that have enabled newer methods to surpass their predecessors. One notable example of such an improvement is Gene Expression Programming (GEP).

GEP is a genotype/phenotype genetic algorithm that blends the traits of Genetic Algorithms and Genetic Programming. Unlike Genetic Programming, where individuals are nonlinear entities of diverse shapes and sizes (parse trees), GEP employs linear strings of fixed length (the genome or chromosomes) as individuals, which are then transformed into nonlinear entities of various shapes and sizes (e.g., simple diagram representations or expression trees), as expounded in [91] (Figure 3.16).

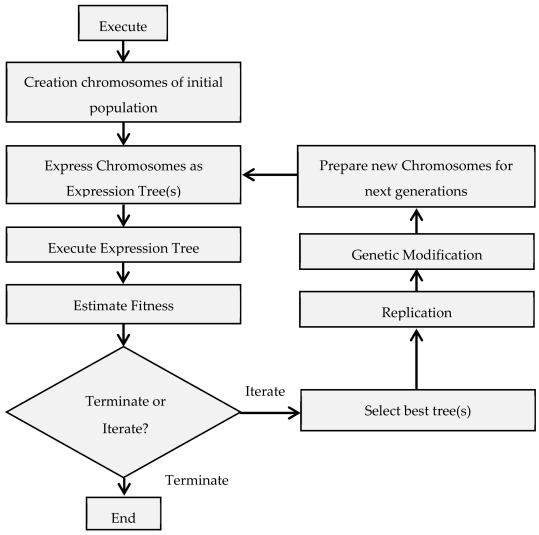

FIGURE 3.16: Working scheme of gene expression programming. Image from [92].

This fundamental distinction empowers GEP to possess a more effective representation compared to GP, as the linear structure used in GEP is more suitable for crossover and mutation operations, leading to faster convergence

and better performance. Coupled with the superior scalability exhibited by GEP, this characteristic sets it apart from previous forms of adaptive algorithms.

Cartesian genetic programming

Cartesian Genetic Programming (CGP) is another newer version of GP, as described in [93]. This approach utilizes a graph representation to encode solutions, rather than the traditional tree or linear genome structure used in GP.

In CGP, solutions are represented as Directed Acyclic Graphs (DAGs), with nodes representing mathematical functions or operators and edges representing connections between the nodes (Figure 3.17). Input variables are also represented as nodes, and the output is generated by evaluating the value of the nodes in the graph. The graph structure in CGP is fixed and determined by a set of parameters such as the number of nodes and the number of inputs and outputs. These parameters can be adjusted to control the size and complexity of the solutions.

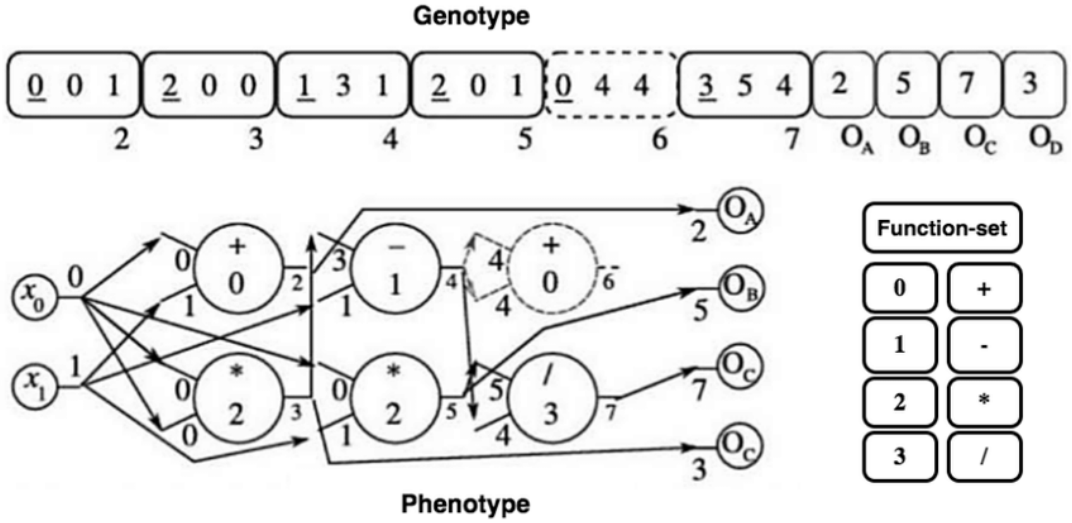

FIGURE 3.17: Working scheme of cartesian genetic programming. Image from [94].

Compared to standard GP, CGP has been shown to outperform in applications requiring the generation of complex programs to solve complex problems. The fixed graph structure, in combination with the ability to control the size and complexity of solutions, makes CGP a powerful tool for solving difficult problems.

3.3.2 Symbolic regression

Symbolic regression is a form of regression analysis that seeks to discover the mathematical expression that best models a given dataset, in terms of both accuracy and simplicity. Various methods, including genetic programming,

Bayesian methods, and neural networks [95], have been employed to tackle the problem of symbolic regression. In this section, we focus solely on the genetic programming-based method.

The process of symbolic regression begins with the creation of initial expressions, which are formed by randomly combining mathematical building blocks, such as operators, functions, constants, and variables (Figure 3.18). Subsequently, new equations are formed by recombining previous ones and probabilistically modifying their subexpressions. The algorithm retains equations that best fit the experimental data and abandons unpromising solutions. Upon achieving the desired level of accuracy, the algorithm terminates, returning a set of equations that are most likely to correspond to the underlying mechanisms governing the observed system. This approach can be used to distil physical and geometrical laws from experimental data, which can be used to bootstrap laws for progressively more complex systems, as demonstrated in [96].

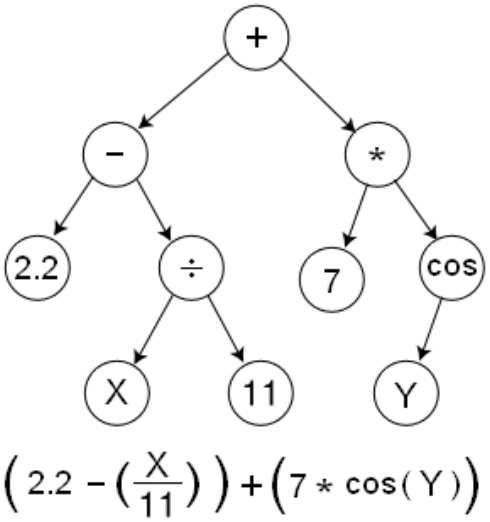

FIGURE 3.18: Expression tree of a symbolic regression algorithm. Image from Wikipedia.

Symbolic regression has numerous advantages over other machine learning techniques. It produces a transparent and interpretable model, thereby providing better insights into the relationship between input and output variables. Moreover, it can identify new mathematical formulas or relationships that can be employed for prediction. Furthermore, symbolic regression can handle missing and noisy data by discovering the best-fit function based on the available data.

The versatility and power of symbolic regression make it a widely used and valuable technique for various applications. For instance, [97] demonstrates how symbolic regression can construct analytic expressions for value functions in reinforcement learning, surpassing other numerical approximators such as neural networks.

Similarly to the case of genetic programming, symbolic regression has been progressively improved over the years, resulting in the development of noteworthy variants.

Multi-objective symbolic regression

Multi-objective symbolic regression, a variant of symbolic regression, aims to find a set of mathematical expressions that can best approximate multiple target functions simultaneously, as discussed in [98]. Unlike traditional symbolic regression, which focuses on finding a single mathematical expression that approximates a single target function, multi-objective symbolic regression considers the trade-offs between different target functions and attempts to identify a set of mathematical expressions that optimizes all of them simultaneously. This approach can be useful in situations where multiple objectives are in conflict with each other, or where there is no clear consensus on which target function is the most important.

Several improvements have been proposed for this technique, including the approach described in [99]. This paper attempts to incorporate prior knowledge about the physical properties of a system into the optimization phase of a symbolic regression algorithm. To achieve this, it employs multi-objective symbolic regression to optimize models with respect to the training accuracy and their compliance with prior knowledge about the properties of the sought model. Additionally, it introduces a phase that generates models by mixing features of models produced in the first phase, and a non-parametric selection method to choose the final model from the entire set of candidates. These advancements highlight the usefulness of multi-objective symbolic regression in various applications, including those with complex, multi-objective systems.

3.3.3 Sparse identification of nonlinear dynamics

Sparse Identification of Nonlinear Dynamics (SINDy) is an equation discovery technique that is particularly well-suited for non-linear dynamical systems. In order to understand how SINDy works, it is first necessary to define the concept of sparse regression.

Sparse regression

Sparse regression [100] is a regression analysis technique that aims to identify the subset of predictor variables that are most relevant in predicting the response variable while minimizing the impact of irrelevant or redundant variables. The goal of this technique is to produce a simple model that predicts the response while minimizing the number of variables used.

This technique assumes that the equations of the underlying model are sparse in the space of possible functions. For example, consider a linear regression model with two predictor variables, x_1 and x_2, and a response variable y. If there is reason to believe that only one of the predictor variables

is actually relevant in predicting the response variable and the other is just noise, then the model is sparse in the space of possible functions.

SINDy

The Sparse Identification of Nonlinear Dynamics (SINDy) is an equation discovery technique developed by Brunton and colleagues [101] that is specifically designed for non-linear dynamical systems. SINDy is a data-driven method used to identify the underlying governing equations of a dynamical system from measured data. The approach employs sparsity-promoting regression algorithms and nonlinear function approximators to identify the most important terms in the underlying dynamical equations, construct a parsimonious model, and capture the essential features of the system's behavior.

SINDy is essentially a combination of symbolic regression and sparse regression techniques. Symbolic regression is utilized to represent the nonlinear functions in the dynamical system equations and to identify a set of candidate functions. These candidate functions are then combined using a sparse regression algorithm to determine the underlying dynamical equations of the system. Nonlinear terms such as polynomials, trigonometric functions, and exponentials are typically included in the candidate functions, and the sparse regression algorithm selects the most relevant terms to construct a parsimonious model of the system dynamics.

Sparse regression enables the SINDy algorithm to surpass standard symbolic regression in terms of scalability to large systems and avoidance of overfitting while maintaining accuracy and model complexity [101]. However, the algorithm may not identify an accurate sparse model when the measurement coordinates and function basis are not amenable to sparse representation. To overcome this limitation, it is highly recommended to incorporate expert knowledge about the physical properties of the system.

Kaiser et al. [102] proposed a method to pair the SINDy model with Model Predictive Control (MPC) to work within the low data limit. The authors extended the SINDy modelling procedure to include the effects of actuation and demonstrated the ability of these models to enhance the performance of MPC based on limited, noisy data. The resulting SINDY-MPC framework offers higher performance, requires significantly less data, and is more computationally efficient and robust to noise than neural network models. This makes it suitable for online training and execution in response to rapid system changes. The paper also suggests using linear models such as the linear Dynamic Mode Decomposition immediately after an abrupt change when the amount of data is still very low until a more accurate SINDy model is trained.

3.4 Chapter summary and conclusion

This chapter presents an overview of various machine-learning techniques that can be utilized for the identification of complex system models.

The discussion commences with a thorough analysis of traditional machine learning approaches, collectively referred to as "classic techniques", which do not involve neural networks or evolutionary algorithms (Section 3.1). Linear regression and its application to non-linear dynamical systems within ARX autoregressive models are described in Subsection 3.1.1, while Support Vector Regression (SVR) is presented in Subsection 3.1.2. Non parametric regression techniques, namely decision tree regression and Gaussian regression, are outlined in Subsections 3.1.3 and 3.1.4, respectively.

Following the discussion on classic techniques, the chapter delves into model learning methods based on deep learning (Section 3.2). After introducing artificial neural networks, Subsection 3.2.1 explains the concept of Recurrent Neural Networks (RNNs) and their two more sophisticated variants, LSTMs and GRUs. Subsection 3.2.2 briefly explores the potential of transformers and their application to physical systems. Subsection 3.2.3 then outlines the advantages of using autoencoders to reduce input data dimensionality, with a focus on convolutional autoencoders and their application in model learning.

The final section, Section 3.3, explores equation discovery algorithms based on evolutionary computations. Subsection 3.3.1 describes Genetic Programming, the foundation of equation discovery techniques, and some of its most significant variants. Subsection 3.3.2 explores the application of GP to mathematical formulas under the name of Symbolic Regression, with a focus on Multi-Objective Symbolic Regression and its advantages. Lastly, Subsection 3.3.3 outlines the even more innovative Sparse Identification of Nonlinear Dynamics (SINDy) technique, which combines sparse regression and symbolic regression.

In summary, this chapter presents various model learning techniques that can be applied to nonlinear dynamical systems and can prove beneficial for the purpose of this review.

Chapter 4

Learning legged robots

Chapter 2 outlined the distinctive features of legged robots, while Chapter 3 delved into various techniques for acquiring models of complex systems. The present chapter seeks to integrate the preceding two by exploring the current state of research on implementing model learning approaches to legged systems, particularly quadrupeds.

4.1 Hybrid systems

Legged systems are unique in terms of their modelling characteristics, and one of their most significant features is their hybrid dynamics [103], which sets them apart from other types of robots. Hybrid dynamics is the combination of continuous and discrete dynamics that govern the motion and behaviour of the system, as depicted in Figure 4.1. The model of these systems can generally be represented in several ways, such as with differential equations including Dirac impulses or modified Petri nets, as discussed in [104].

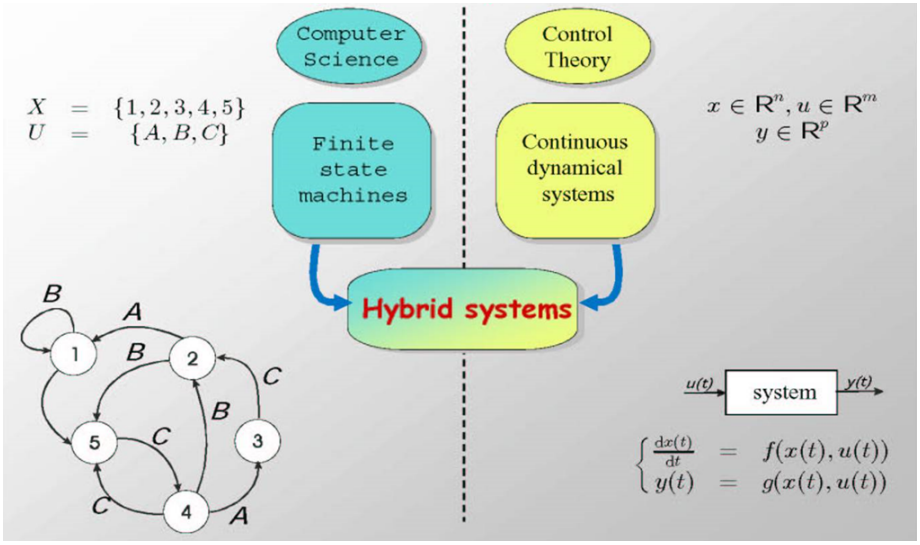

FIGURE 4.1: Conceptual scheme of hybrid systems.

The continuous dynamics of a legged robot involve the motion of its joints, actuators, and other mechanical components, which determine the robot's movement and interaction with its environment in a continuous manner. The discrete dynamics refer to the discrete events that occur during the system's motion, such as the switch of contact with the ground, which involves a change in the robot's support configuration and can significantly impact the robot's motion and stability. The different gaits that a legged system can employ, such as walking, running or hopping, and the transitions between them are also part of the discrete dynamics.

The hybrid dynamics of a legged system result from the combination of these continuous and discrete dynamics. During a walking gait, for example, the continuous dynamics govern the motion of the system's joints and actuators as it moves its legs, while the discrete dynamics determine the timing and sequence of the leg movements.

In the following sections, various methods to incorporate this hybrid behaviour into the model of a legged system will be explored.

4.1.1 Hybrid Zero Dynamics model

One viable method for managing the modelling and control of hybrid systems involves the use of a Hybrid Zero Dynamics (HZD) model. The objective is to devise a control law that guarantees the convergence of the output trajectory of the system to a desired equilibrium or set point, while also ensuring that specific state variables remain zero or reach zero within a finite time frame. These state variables, known as "hybrid zeros", correspond to the system's switching points or discontinuities in behaviour.

Initially developed for biped robots [105], the HZD model has been extended to other legged robots, owing to its ability to provide a hybrid model of locomotion with invariant zero dynamics manifolds under both continuous and discrete time dynamics [106]. In contrast to single-body models, the HZD model accounts for the interaction between the internal states of the robot and its contact forces with the environment, resulting in a more precise prediction of the robot's behaviour in complex situations.

The HZD approach overcomes the limitations of conventional control approaches by decomposing the system's dynamics into two parts: a "zero dynamics" part that captures the system's intrinsic stability properties, and a "nonzero dynamics" part that captures the system's agility and manoeuvrability. By designing controllers for the zero dynamics part first, it is possible to ensure stable locomotion while still allowing for agile movements and manoeuvres.

While the HZD modelling approach is a powerful technique for handling hybrid systems, it is worth noting that it produces a simplified model that captures the fundamental dynamics of the robot's locomotion. The next subsection will focus on machine learning methods for actually learning the dynamic model of hybrid systems.

4.1.2 Learning hybrid systems

Although traditional mathematical modelling techniques can be used to construct models for hybrid dynamical systems, as exemplified in [107], this subsection primarily focuses on the methods that rely on experimental data.

Hybrid system identification

As described in [108], the use of first principles modelling to construct models for hybrid dynamical systems is often too complex or unfeasible in practical situations, necessitating the identification of models from experimental data. To this end, the paper presents four distinct approaches for defining the model of a hybrid system.

Of these approaches, the paper focuses on two types of models for hybrid systems: the switched affine model and the piecewise affine model. The former comprises collections of linear/affine models interconnected by switches indexed by a discrete-valued additional variable, known as the discrete state. In this category, the paper specifically examines the Switched affine AutoRegressive eXogenous (SARX) model, which resembles autoregressive models discussed in Subsection 3.1.1. The latter model, the piecewise affine model, is similar to the switched affine model, but the discrete state is determined by a polyhedral partition of the state-input domain, as illustrated in Figure 4.2. This model can also be defined as an autoregressive model, known as the PieceWise affine AutoRegressive eXogenous (PWARX) model.

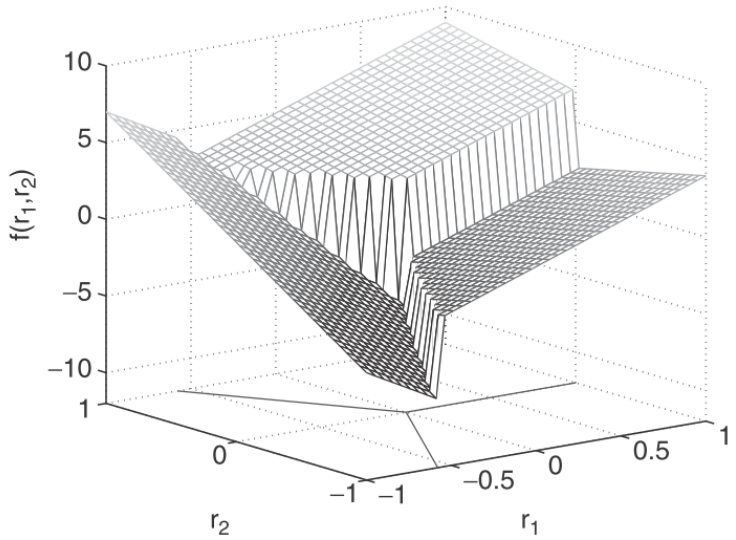

FIGURE 4.2: Discontinuous PWA map of two variables. Image from [108].

The paper then explores different techniques for identifying these models from a hybrid system. The Algebraic Procedure treats the identification of SARX models as an algebraic geometric problem and offers a closed-form solution to the identification problem that is provably correct in the absence of noise. The Clustering-Based Procedure, in contrast, utilizes clustering, linear identification, and pattern recognition techniques to identify

a potentially discontinuous piecewise affine map with a multi-dimensional domain. The Bayesian procedure takes advantage of available prior knowledge about the modes and parameters of the hybrid system by modelling parameter vectors as random variables described by their probability density functions. Lastly, the Bounded-Error Procedure leverages optimization-based techniques to minimize the difference between the actual system behaviour and the predicted behaviour, subject to a bounded error constraint.

Overall, the paper concludes that each procedure is suited to specific situations and has its own advantages and limitations. It suggests that combining the features of these procedures could improve their effectiveness. The paper also notes that several other models, such as Mixed Logical Dynamical (MLD) models, Linear Complementarity (LC) models, Extended Linear Complementarity (ELC) models, and Max-Min-Plus-Scaling (MMPS) models, as well as various mathematical, statistical, and machine-learning techniques, can be employed to identify the model of hybrid systems. An excellent review of such techniques can be found in [109].

The next paragraph will present one of the aforementioned methods with particular relevance for this review.

Multi-modal symbolic regression

Ly et al. [110] proposed a method for addressing the problem of identifying hybrid systems using symbolic regression, discussed in Subsection 3.3.2.

The authors used the hybrid automata model to represent the system, which is defined as a 5-tuple $\mathcal{H} = (\mathcal{W}, \mathcal{X}, \mathcal{M}, \mathcal{F}, \mathcal{T})$. Here, $\mathcal{W}$ represents the communication space for external variables, $\mathcal{X}$ denotes the continuous space for continuous state variables, $\mathcal{M}$ represents the countable, discrete set of modes, $\mathcal{F}$ is the countable, discrete set of first-order, coupled, differential-algebraic equations, and $\mathcal{T}$ is the countable, discrete set of transitions or events. The hybrid automata's latent state is determined by both the discrete mode and the continuous state space vector. For example, in Figure 4.3, $\mathcal{W}$ comprises the two inputs u_1 and u_3, $\mathcal{X}$ comprises the state variables x and x', $\mathcal{M}$ consists of three modes $\{m_1, m_2, m_3\}$ which represent distinct vehicle behaviours, $\mathcal{F}$ consists of $\{f_1, f_2, f_3\}$ and describes these behaviours, and $\mathcal{T}$ consists of five Boolean conditions, each representing a transition event.

The paper then proceeds to explain the algorithm that will be used for the learning problem, called multi-modal symbolic regression. The algorithm is made by two sub-algorithms: Clustered Symbolic Regression (CSR) and Transition Modeling (TM).

In the case of Clustered Symbolic Regression, the problem is defined as finding a model that minimizes the within-domain absolute error of a piecewise function, where each subdomain is represented by a symbolic expression. The problem is solved using unsupervised learning techniques, where the algorithm infers a symbolic model for each subdomain while distinguishing individual functions. The algorithm combines two learning approaches: Symbolic Regression (SR) and Expectation Maximization (EM). While SR was

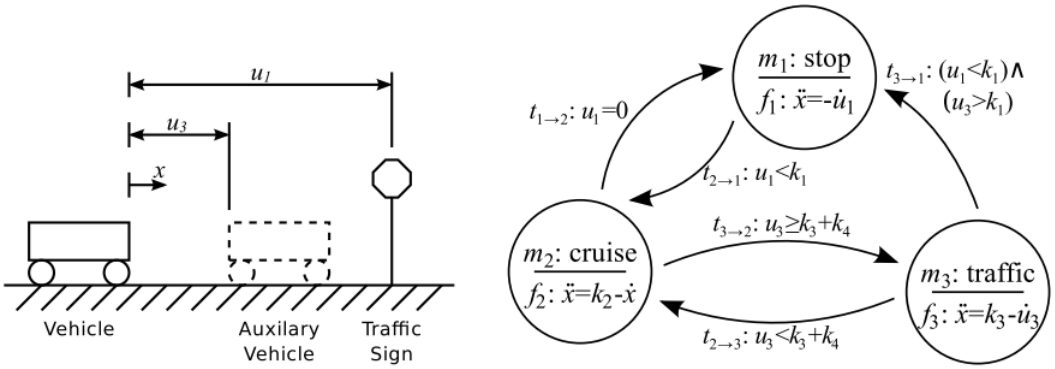

FIGURE 4.3: Example of a hybrid automata model for a 1D driverless car. Image from [110].

already discussed in Subsection 3.3.2, EM is a statistical algorithm that iteratively estimates the parameters of a statistical model. In the CSR algorithm, EM is used to cluster the data into subdomains based on their input-output pairs' probabilities. The algorithm alternates between clustering the data and inferring the symbolic expressions for each subdomain until convergence. Overall, CSR can learn nonlinear and complex expressions, including rational equations, and can be used to model multi-modal systems.

On the other hand, Transition Modeling is a supervised learning algorithm used to determine symbolic, discriminant inequalities for transition events by reformulating a classification problem as a regression problem using function composition. The problem TM solves is a binary classification problem, where an input vector at index n belongs to a characteristic domain, and its corresponding label is 1 or 0, respectively. The discriminant function that describes the characteristic domain and minimizes the classification error is inferred using the TM algorithm. It builds on the infrastructure of SR and the discriminant functions are expressed symbolically as an inequality where the data has membership if the inequality evaluates to true. TM uses SR to generate temporary solutions and calculates temporary membership values and variance for each mode. It then computes the global fitness using temporary values and returns behaviours and variances. Although evolutionary computation for classification has been previously investigated, TM is unique in that it adapts the well-developed framework of SR, allowing for a unified approach to both domains.

Finally, the multi-modal symbolic regression algorithm (MMSR) combines the two sub-algorithms in order to determine the modes, behaviours, and transitions of the hybrid system. The first step is to apply CSR to cluster the data into distinct modes while simultaneously inferring symbolic expressions for each sub-function. Using the modal memberships from CSR, TM is then applied to find symbolic expressions for the transition conditions. Finally, the best expression is selected as the one with the highest fitness metric.

Overall, MMSR outperformed its counterparts in terms of model accuracy and complexity, and successfully identified and characterized field-effect transistor modes. The paper concludes that symbolic modelling has numerous benefits over parametric numerical models, and the MMSR algorithm

presents a viable alternative approach that may have the additional benefit of insight and interpretability.

4.2 Existing work on quadrupeds

A majority of the literature focused on the control of quadruped robots employs reduced-order template models, as delineated in Subsection 2.2.2. These models are efficacious for evaluating control algorithms as they are both user-friendly and perceptive regarding dynamic behaviour. However, they are inherently limited and do not possess the requisite complexity to apply more complex control techniques. Conversely, the utilization of full-order models, when available, is prohibitively computationally expensive due to their high dimensionality.

Several endeavours have attempted to bridge the gap between reduced- and full-order models through the use of control algorithms. For instance, [111] presents a layered control approach that generates optimal trajectories for an SRB reduced-order model and subsequently utilizes a nonlinear controller to map the optimal reduced-order trajectories onto the full-order model. In contrast, [112] introduces analogous work with a LIP reduced-order model.

However, these control solutions are merely a means of circumventing the problem rather than successfully consolidating the advantages of both reduced- and full-order models into a single type of model. Recent literature has made some attempts to address this issue, and the following paragraphs will present a few examples.

4.2.1 Parameter identification

One methodology that can be employed to construct models of complex systems is parameter identification [113]. It is a technique that utilizes measured data to learn the values of unknown parameters in a pre-specified dynamic model of a system. The objective of parameter identification is to determine the values of these parameters that best fit the measured data, typically by minimizing the difference between the predicted and measured data using an optimization algorithm.

The pre-specified dynamic model may be a full-order model that captures all of the details of the robot's dynamics or a reduced-order model that captures the essential features of the robot's dynamics using a smaller number of parameters. In other words, this technique can exploit template models to create fine-tuned reduced-order models for quadrupeds. However, the aforementioned limitations still apply depending on the choice of the pre-specified model.

A fascinating example of the application of this technique to quadruped robots is presented in [114]. The study addresses the challenge of incorporating additional robotic hardware to a commercial quadruped robot, such as Boston Dynamic's Spot. As indicated in Subsection 2.2.1, the company does

not disclose any documentation on their robots, and hence their low-level workings are to be regarded as black-box. Therefore, when implementing a robotic arm on the robot, as illustrated in Figure 4.4, the model of the quadruped must be re-learned to combine it with the arm to control the full system.

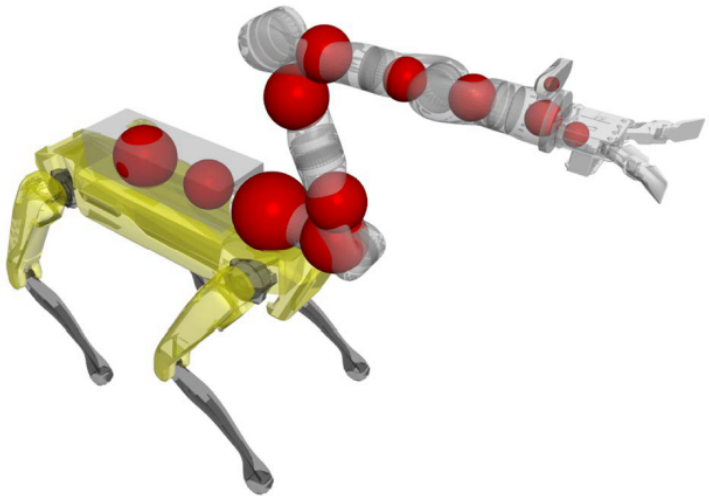

FIGURE 4.4: Graphical representation of a Spot robot with a robotic arm on top. Image from [114].

As a pre-specified model, the paper uses a simple kinetic arm with a floating base. The state is represented by the stacked position and orientation of the body and the arm's joint angles, while the control input represents the velocity commands sent to the individual platforms. Thus, the basic dynamical system is expressed as:

$$x_{i+1} = x_i + h \cdot u_i$$

where x is the state, u is the control input, i is the time step, and h is the time step size. This simple model appears to approximate well the behavior of the body and the arm when they are operated independently, but it fails when combined, owing to the additional forces that the arm applies to the body.

In fact, without any payload updates, Spot starts to drift when the arm is moved, causing it to deviate from its intended path. For this reason, the authors try to correct the model using an unknown correction term $\mathcal{B}$ to reduce the discrepancy between the nominal state $\bar{x}_{nom}$ and the actual one, as well as to express the interaction between the body and the arm using the current and past positions of the CoM of the arm. The pre-specified model thus becomes:

$$x_{i+1} = x_i + h \cdot u_i + \mathcal{B}(x_i, x_{i-1}, x_{i-2})$$

The authors then employ parameter identification to this reduced-order model to find the optimal parameters.

The results obtained using this method are impressive and succeed in

building a suitable model of the robot for model-based control. It is noteworthy to mention the procedure of commencing with a simple model, upgrading it to meet the requirements, and then using real data to fine-tune its parameters.

The authors discuss how this technique succeeds in achieving a satisfactory outcome, but it has limitations with respect to disturbances and complex behaviour. They mention the possibility of using learning-based approaches, which necessitate significantly more data and higher computational time but can produce much more precise and efficient models. The following subsection will address this topic.

4.2.2 Learning-based approaches

The existing academic literature reveals a scarcity of research studies pertaining to learning-based approaches that are exclusively designed for developing models of quadruped robots. As stated earlier, the majority of the available studies either utilize pre-existing reduced-order models or physics-based full-order models. There have been a limited number of attempts to leverage machine learning techniques for developing legged robot models, and among them, Gaussian Processes have emerged as the most widely adopted solution.

Model-based reinforcement learning for hexapod robot

In an effort to apply model-based reinforcement learning to a hexapod robot, Chatzilygeroudis et al. [115] employed a learning-based approach to model the system, specifically Gaussian processes (outlined in Subsection 3.1.4). In contrast to traditional Gaussian regression, the authors incorporated prior knowledge (Figure 4.5) into the process to improve model accuracy, particularly in regions with sparse or absent real-world data.

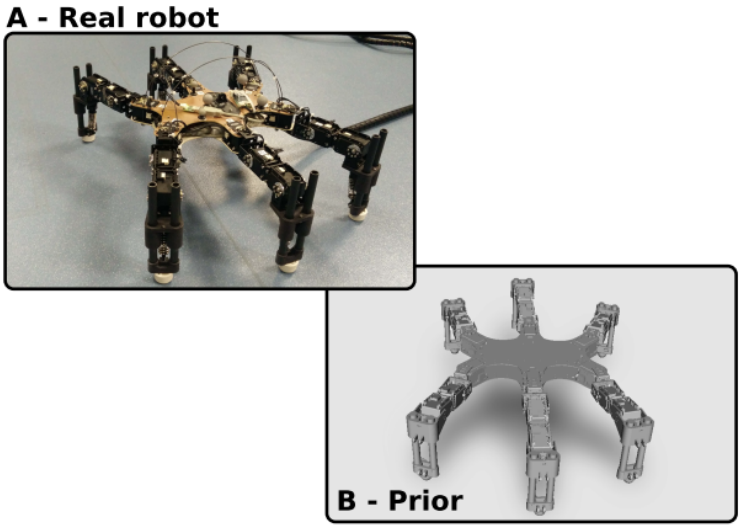

FIGURE 4.5: Physical robot and its simulated version used as a prior. Image from [115].

To incorporate prior knowledge into the model learning procedure using Gaussian processes, the authors introduced priors by assuming that the prior model M was a Gaussian process learned from simulation data gathered from running the model on the prior system. The prior model M was defined as a tunable or non-tunable mean function, with the hyperparameters of the kernel used in the Gaussian process learned via Maximum Likelihood Estimation (MLE) by maximizing the likelihood function of the training data. Consequently, priors were integrated into the learning procedure using a Gaussian process with a prior mean function obtained from simulation data.

By combining the prior model with real-world data using Gaussian processes, the model was able to quickly adapt to new situations and make accurate predictions. In addition, prior knowledge helped to reduce the uncertainty in the model predictions, which was especially crucial in reinforcement learning tasks, where inaccurate predictions could lead to suboptimal performance.

Data-Enabled Predictive Control for a quadruped robot

In the quest for tailored approaches for quadrupeds, a recent work presented in [116] introduces Data-Enabled Predictive Control (DeePC), a form of Data-Driven Predictive Control. This method differs from standard Model Predictive Control (MPC) by constructing the model directly from observed trajectories, as opposed to relying on a physics-based model.

To construct the model, the authors utilize Behavioural Systems Theory (BST), which parameterizes a Linear Time-Invariant (LTI) system in terms of its input-output behaviour, as shown in Figure 4.6. The system dynamics are represented using impulse responses, obtained by applying impulses at different times, which are combined to create a linear time-invariant model that predicts the system's response to any input.

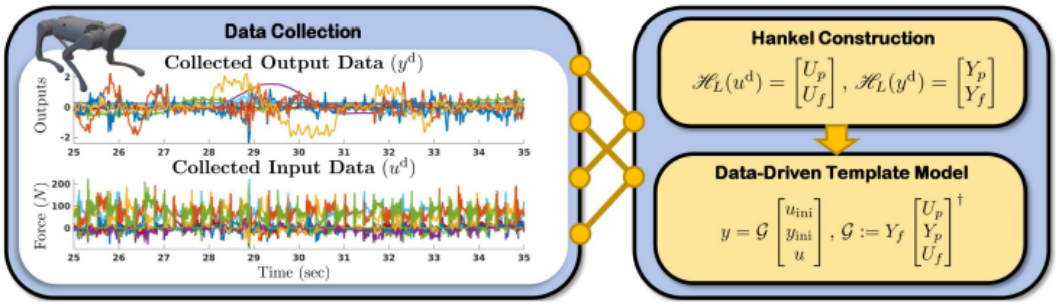

FIGURE 4.6: Overview of the process used to construct the data-driven template model. Image from [116].

While this paper does not employ any of the machine learning techniques discussed previously, it is important to mention the trajectory clustering approach used here. This method is not considered a machine learning approach, but rather a form of data analysis. It involves clustering similar system trajectories based on their Euclidean distances and then fitting a linear

model to each cluster. This allows for the creation of a compact and interpretable model that can be used for control, even in the absence of a complete physics-based model of the system.

4.3 Chapter summary and conclusion

This chapter merges the preceding two chapters and delineates the application of model learning techniques to legged systems.

Section 4.1 addresses the challenge of hybrid systems, initially explicating their nature and subsequently addressing methods of handling them. The Hybrid Zero Dynamics approach described in Subsection 4.1.1 presents a promising means of handling hybrid systems using a reduced-order model. However, the crux of this section resides in Subsection 4.1.2, which first enumerates various hybrid system identification techniques and then concentrates on multi-modal symbolic regression.

The second part of this chapter, Section 4.2, is focused more narrowly on quadrupeds and the works done in this specific sub-category of legged systems. Subsection 4.2.1 probes the parameter identification approach, particularly in the context of a Spot robot equipped with a robotic arm mounted on top. In contrast, Subsection 4.2.2 scrutinizes learning-based methods for constructing models of quadrupeds. As a result of the paucity of results in this area, it delineates a technique based on Gaussian Processes that was utilized to learn the model of a hexapod robot, as well as a data analysis technique based on Behavioural Systems Theory, which was employed to learn the model of a quadruped robot.

In summary, this chapter endeavours to analyze the existing literature regarding the model learning of quadruped robots. Ultimately, it highlights a lack of studies pertaining to machine-learning-based techniques applied to this particular task.

Chapter 5

Conclusion

5.1 Discussion

5.1.1 State of the art of quadruped models

This review has examined various studies conducted on quadruped robots with the aim of comprehending the authors' approaches to addressing the modelling challenge.

Certain studies begin by constructing uncomplicated quadruped robots, for which obtaining the full-order model is relatively straightforward. This facilitates precise control of the system, without incurring a high computational expense, owing to the robot's relative simplicity.

However, most studies utilize preexisting commercial robots, such as Boston Dynamics' Spot. In such cases, the full-order model is unavailable and obtaining it through conventional modelling techniques would prove exceedingly challenging. Furthermore, deploying full-order models of such intricate systems would be extremely computationally inefficient when implementing control methods. Consequently, these studies tend to rely on reduced-order template models, such as the Single Rigid Body (SRB) or the Spring-Loaded Inverted Pendulum (SLIP). Although this approach furnishes a simple model that is easy to control, it tends to overlook certain minor dynamic aspects that hinder the attainment of accurate control performance.

As far as the author is aware, no noteworthy research has been conducted that employs an intermediate approach between the two aforementioned methods: a framework that encompasses the majority of the system's dynamics while being optimized to avoid computational complexity. To put it differently, a lower-dimensional dynamic model that is custom-designed for the particular quadruped under investigation.

5.1.2 State of the art of model-learning techniques

This paper has explored various techniques for learning models of complex systems, each with its unique advantages and drawbacks.

Gaussian regression has emerged as the most promising classic technique for modelling highly non-linear dynamic systems, such as legged robots. This approach can capture uncertainties, a critical feature when dealing with disturbances. Moreover, Gaussian regression is more data-efficient than other traditional methods.

In terms of deep learning techniques, recurrent neural networks have proven to be effective in capturing temporal dependencies in data, while autoencoders are useful for reducing the dimensionality of data by selecting the most important features.

In contrast, equation discovery techniques based on evolutionary algorithms have gained popularity due to their ability to produce interpretable models. Despite requiring more data and time to train than neural networks, these methods offer much better interpretability in the resulting models, making them highly relevant for control applications.

Application to quadrupeds

In regards to the implementation of model learning techniques in the context of quadruped robots, the existing literature is notably sparse and provides only a handful of examples that are not entirely pertinent.

Of all the aforementioned techniques, symbolic regression appears to hold significant promise. The interpretability of the resulting models would prove particularly advantageous for control applications. Additionally, multi-modal symbolic regression has proven to be highly effective in modelling hybrid systems, as detailed in the review. Some variations of symbolic regression may also prove useful in this specific case, such as multi-objective symbolic regression, which could be utilized when prior knowledge regarding the physical properties of the system is available. SINDy presents another significant possibility, as it combines the advantages of symbolic regression with the sparsity analysis of sparse regression.

Moreover, several examples have demonstrated that the use of autoencoders can be combined with the actual model learning technique and it can prove particularly useful with complex systems that possess high input data dimensionality.

5.2 Thesis problem statement

The present review has revealed a noteworthy gap in the literature pertaining to the acquisition of a reduced-order dynamic model for quadruped robots, which could be employed in numerous subsequent control applications. The proposed thesis aims to address this gap by adopting the approach suggested in the discussion.

Initially, input-output data from a simulated quadruped robot will be collected, following which an autoencoder will be trained on the data until an encoder capable of effectively reducing the dimensionality of the data is developed. Subsequently, multiple equation discovery techniques will be evaluated and combined in an attempt to identify the most optimal solution. Of particular relevance, as emphasized in the review, are the sparsity-based approach of SINDy, the ability to optimize for prior physical knowledge of multi-objective symbolic regression, and the capability to manage the hybrid nature of the system of multi-modal symbolic regression. The objective is to integrate these features to the greatest extent possible, to identify an equation

discovery method best suited for this specific purpose. Finally, the encoder and the enhanced symbolic regression algorithm will be tested on actual quadruped robots to verify their generalization capabilities and assess their ability to consistently produce a reduced-order dynamic model for quadruped robots.

Bibliography

[1] A. Liberati, D. G. Altman, J. Tetzlaff, *et al.*, "The prisma statement for reporting systematic reviews and meta-analyses of studies that evaluate health care interventions: Explanation and elaboration", *Annals of internal medicine*, 2009.

[2] M. F. Silva and J. T. Machado, "A literature review on the optimization of legged robots", *Journal of Vibration and Control*, 2012.

[3] S. Kim, P. M. Wensing, *et al.*, "Design of dynamic legged robots", *Foundations and Trends® in Robotics*, 2017.

[4] P. Holmes, R. J. Full, D. Koditschek, and J. Guckenheimer, "The dynamics of legged locomotion: Models, analyses, and challenges", *SIAM review*, 2006.

[5] D. J. Todd, *Walking machines: an introduction to legged robots*. Springer Science & Business Media, 2013.

[6] S. Böttcher, "Principles of robot locomotion", in *Proceedings of human robot interaction seminar*, 2006.

[7] P. Biswal and P. K. Mohanty, "Development of quadruped walking robots: A review", *Ain Shams Engineering Journal*, 2021.

[8] M. H. Raibert, "Legged robots", *Communications of the ACM*, 1986.

[9] S. Hirose and H. Takeuchi, "Study on roller-walk (basic characteristics and its control)", in *Proceedings of IEEE International Conference on Robotics and Automation*, IEEE, 1996.

[10] M. Bjelonic, P. K. Sankar, C. D. Bellicoso, H. Vallery, and M. Hutter, "Rolling in the deep–hybrid locomotion for wheeled-legged robots using online trajectory optimization", *IEEE Robotics and Automation Letters*, 2020.

[11] U. Saranli, M. Buehler, and D. E. Koditschek, "Rhex: A simple and highly mobile hexapod robot", *The International Journal of Robotics Research*, 2001.

[12] T. J. Allen, R. D. Quinn, R. J. Bachmann, and R. E. Ritzmann, "Abstracted biological principles applied with reduced actuation improve mobility of legged vehicles", in *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)(Cat. No. 03CH37453)*, IEEE, 2003.

[13] R Mosher, "Test and evaluation of a versatile walking truck", in *Proceedings of Off-Road Mobility Research Symposium, Washington DC, 1968*, 1968.

[14] A. A. Frank, *Automatic control systems for legged locomotion machines*. University of Southern California, 1968.

[15] S. Hirose and K. Kato, "Study on quadruped walking robot in tokyo institute of technology-past, present and future", in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, IEEE, 2000.

[16] M. H. Raibert, *Legged robots that balance*. MIT press, 1986.

[17] K. Arikawa and S. Hirose, "Development of quadruped walking robot titan-viii", in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems. IROS'96*, IEEE, 1996.

[18] H Miura, I Shimoyama, M Mitsuishi, and H Kimura, "Dynamical walk of quadruped robot (collie-1)", in *Int. Symp. Robotics Research*, Cambridge, MA: MIT Press, 1985.

[19] H. Kimura and Y. Fukuoka, "Biologically inspired adaptive dynamic walking in outdoor environment using a self-contained quadruped robot: 'tekken2'", in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*, IEEE, 2004.

[20] H. Kimura, Y. Fukuoka, and K. Konaga, "Adaptive dynamic walking of a quadruped robot using a neural system model", *Advanced Robotics*, 2001.

[21] M. Buehler, R. Battaglia, A. Cocosco, G. Hawker, J. Sarkis, and K. Yamazaki, "Scout: A simple quadruped that walks, climbs, and runs", in *Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No. 98CH36146)*, IEEE, 1998.

[22] J. G. Nichol, S. P. Singh, K. J. Waldron, L. R. Palmer Iii, and D. E. Orin, "System design of a quadrupedal galloping machine", *The International Journal of Robotics Research*, 2004.

[23] W. Ilg, J. Albiez, H Jedele, K. Berns, and R. Dillmann, "Adaptive periodic movement control for the four legged walking machine bisam", in *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No. 99CH36288C)*, IEEE, 1999.

[24] P. G. De Santos, E. Garcia, and J. Estremera, *Quadrupedal locomotion: an introduction to the control of four-legged robots*. Springer, 2006.

[25] X. Rong, Y. Li, J. Ruan, and B. Li, "Design and simulation for a hydraulic actuated quadruped robot", *Journal of mechanical science and technology*, 2012.

[26] M. Li, Z. Jiang, P. Wang, L. Sun, and S. S. Ge, "Control of a quadruped robot with bionic springy legs in trotting gait", *Journal of Bionic Engineering*, 2014.

[27] M. Raibert, K. Blankespoor, G. Nelson, and R. Playter, "Bigdog, the rough-terrain quadruped robot", *IFAC Proceedings Volumes*, 2008.

[28] C. Semini, N. G. Tsagarakis, E. Guglielmino, M. Focchi, F. Cannella, and D. G. Caldwell, "Design of hyq–a hydraulically and electrically actuated quadruped robot", *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, 2011.

[29] C. Semini, V. Barasuol, J. Goldsmith, *et al.*, "Design of the hydraulically actuated, torque-controlled quadruped robot hyq2max", *IEEE/Asme Transactions on Mechatronics*, 2016.

[30] M. Hutter, C. Gehring, M. Bloesch, M. A. Hoepflinger, C. D. Remy, and R. Siegwart, "Starleth: A compliant quadrupedal robot for fast, efficient, and versatile locomotion", in *Adaptive Mobile Robotics*, World Scientific, 2012.

[31] M. Hutter, C. Gehring, D. Jud, *et al.*, "Anymal-a highly mobile and dynamic quadrupedal robot", in *2016 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, IEEE, 2016.

[32] S. Seok, A. Wang, M. Y. Chuah, D. Otten, J. Lang, and S. Kim, "Design principles for highly efficient quadrupeds and implementation on the mit cheetah robot", in *2013 IEEE International Conference on Robotics and Automation*, IEEE, 2013.

[33] H.-W. Park, P. M. Wensing, and S. Kim, "High-speed bounding with the mit cheetah 2: Control design and experiments", *The International Journal of Robotics Research*, 2017.

[34] G. Bledt, M. J. Powell, B. Katz, J. Di Carlo, P. M. Wensing, and S. Kim, "Mit cheetah 3: Design and control of a robust, dynamic quadruped robot", in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2018.

[35] B. Katz, J. Di Carlo, and S. Kim, "Mini cheetah: A platform for pushing the limits of dynamic quadruped control", in *2019 international conference on robotics and automation (ICRA)*, IEEE, 2019.

[36] R. Featherstone, *Rigid body dynamics algorithms*. Springer, 2014.

[37] V. Vasilopoulos, K. Machairas, and E. Papadopoulos, "Quadruped pronking on compliant terrains using a reaction wheel", in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2016.

[38] H.-W. Kim, S. Hong, and J.-s. Choi, "Comparative study on tracked vehicle dynamics on soft soil: Single-body dynamics vs. multi-body dynamics", in *Fifth ISOPE Ocean Mining Symposium*, OnePetro, 2003.

[39] L. Righetti, J. Buchli, M. Mistry, and S. Schaal, "Inverse dynamics control of floating-base robots with external constraints: A unified view", in *2011 IEEE international conference on robotics and automation*, IEEE, 2011.

[40] M. Mistry, J. Buchli, and S. Schaal, "Inverse dynamics control of floating base systems using orthogonal decomposition", in *2010 IEEE international conference on robotics and automation*, IEEE, 2010.

[41] R. J. Full and D. E. Koditschek, "Templates and anchors: Neuromechanical hypotheses of legged locomotion on land", *Journal of experimental biology*, 1999.

[42] J. Shen and D. Hong, "Convex model predictive control of single rigid body model on so (3) for versatile dynamic legged motions", in *2022 International Conference on Robotics and Automation (ICRA)*, IEEE, 2022.

[43] Y. Ding, A. Pandala, and H.-W. Park, "Real-time model predictive control for versatile dynamic motions in quadrupedal robots", in *2019 International Conference on Robotics and Automation (ICRA)*, IEEE, 2019.

[44] D. E. Orin, A. Goswami, and S.-H. Lee, "Centroidal dynamics of a humanoid robot", *Autonomous robots*, 2013.

[45] T. Kwon, Y. Lee, and M. Van De Panne, "Fast and flexible multilegged locomotion using learned centroidal dynamics", *ACM Transactions on Graphics (TOG)*, 2020.

[46] A. W. Winkler, C. D. Bellicoso, M. Hutter, and J. Buchli, "Gait and trajectory optimization for legged systems through phase-based end-effector parameterization", *IEEE Robotics and Automation Letters*, 2018.

[47] S. Kajita, F. Kanehiro, K. Kaneko, K. Yokoi, and H. Hirukawa, "The 3d linear inverted pendulum mode: A simple modeling for a biped walking pattern generation", in *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No. 01CH37180)*, IEEE, 2001.

[48] A. Iqbal, S. Veer, and Y. Gu, "Drs-lip: Linear inverted pendulum model for legged locomotion on dynamic rigid surfaces", 2022.

[49] K. Theofanous, "Dynamic walkng of legged machines", *CoRR*, 2018.

[50] I. Poulakakis and J. W. Grizzle, "The spring loaded inverted pendulum as the hybrid zero dynamics of an asymmetric hopper", *IEEE Transactions on Automatic Control*, 2009.

[51] M. Shahbazi, R. Babuška, and G. A. Lopes, "Unified modeling and control of walking and running on the spring-loaded inverted pendulum", *IEEE Transactions on Robotics*, 2016.

[52] H Khan, C Semini, D. Caldwell, and V Barasuol, "Actuator sizing for highly-dynamic quadruped robots based on squat jumps and running trots", in *Nature-Inspired Mobile Robotics*, World Scientific, 2013.

[53] D. Maulud and A. M. Abdulazeez, "A review on linear regression comprehensive in machine learning", *Journal of Applied Science and Technology Trends*, 2020.

[54] Z. Zhang, Y. Li, L. Li, Z. Li, and S. Liu, "Multiple linear regression for high efficiency video intra coding", in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2019.

[55] M. Jansson, "Subspace identification and arx modeling", *IFAC Proceedings Volumes*, 2003.

[56] R. T. Baillie, "Predictions from armax models", *Journal of Econometrics*, 1980.

[57] S. Sekizawa, S. Inagaki, T. Suzuki, *et al.*, "Modeling and recognition of driving behavior based on stochastic switched arx model", *IEEE Transactions on Intelligent Transportation Systems*, 2007.

[58] B. M. Achsan, *Support vector machine: Regression*, 2019.

[59] A. J. Smola and B. Schölkopf, "A tutorial on support vector regression", *Statistics and computing*, 2004.

[60] S Kavitha, S Varuna, and R Ramya, "A comparative analysis on linear regression and support vector regression", in *2016 online international conference on green engineering and technologies (IC-GET)*, IEEE, 2016.

[61] Y. Cheng, L. Wang, and J. Hu, "Quasi-arx wavelet network for svr based nonlinear system identification", *Nonlinear Theory and Its Applications, IEICE*, 2011.

[62] G. K. Tso and K. K. Yau, "Predicting electricity energy consumption: A comparison of regression analysis, decision tree and neural networks", *Energy*, 2007.

[63] M. S. Acharya, A. Armaan, and A. S. Antony, "A comparison of regression models for prediction of graduate admissions", in *2019 international conference on computational intelligence in data science (ICCIDS)*, IEEE, 2019.

[64] C. E. Rasmussen and C. K. Williams, "Gaussian processes in machine learning", *Lecture notes in computer science*, 2004.

[65] F. Leclercq, "Bayesian optimization for likelihood-free cosmological inference", *Physical Review D*, 2018.

[66] G. Pillonetto, A. Chiuso, and G. De Nicolao, "Prediction error identification of linear systems: A nonparametric gaussian regression approach", *Automatica*, 2011.

[67] J. Wang, A. Hertzmann, and D. J. Fleet, "Gaussian process dynamical models", *Advances in neural information processing systems*, 2005.

[68] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning", *nature*, 2015.

[69] L. Dormehl, "What is an artificial neural network? here's everything you need to know", *Digital Trends*, 2019.

[70] L. Medsker and L. C. Jain, *Recurrent neural networks: design and applications*. CRC press, 1999.

[71] J Roell, "Understanding recurrent neural networks: The preferred neural network for time series data", *Article in towards data science*, 2017.

[72] E. Grinke, C. Tetzlaff, F. Wörgötter, and P. Manoonpong, "Synaptic plasticity in a recurrent neural network for versatile and adaptive behaviors of a walking robot", *Frontiers in neurorobotics*, 2015.

[73] S. Hochreiter and J. Schmidhuber, "Long short-term memory", *Neural computation*, 1997.

[74] M. Phi, "Illustrated guide to lstm's and gru's: A step by step explanation", *Towards Data Science*, 2018.

[75] X. Yuan, L. Li, and Y. Wang, "Nonlinear dynamic soft sensor modeling with supervised long short-term memory network", *IEEE transactions on industrial informatics*, 2019.

[76] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches", *arXiv preprint arXiv:1409.1259*, 2014.

[77] Y. Zhang and L. Yang, "A novel dynamic predictive method of water inrush from coal floor based on gated recurrent unit model", *Natural Hazards*, 2021.

[78] H. Xie, H. Tang, and Y.-H. Liao, "Time series prediction based on narx neural networks: An advanced approach", in *2009 International conference on machine learning and cybernetics*, IEEE, 2009.

[79] T. Lin, B. G. Horne, P. Tino, and C. L. Giles, "Learning long-term dependencies in narx recurrent neural networks", *IEEE Transactions on Neural Networks*, 1996.

[80] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, "Attention is all you need", *Advances in neural information processing systems*, 2017.

[81] N. Geneva and N. Zabaras, "Transformers for modeling physical systems", *Neural Networks*, 2022.

[82] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks", *science*, 2006.

[83] K. Lee and K. T. Carlberg, "Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders", *Journal of Computational Physics*, 2020.

[84] X. Guo, X. Liu, E. Zhu, and J. Yin, "Deep clustering with convolutional autoencoders", in *Neural Information Processing: 24th International Conference, ICONIP 2017, Guangzhou, China, November 14-18, 2017, Proceedings, Part II 24*, Springer, 2017.

[85] J. Xu and K. Duraisamy, "Multi-level convolutional autoencoder networks for parametric prediction of spatio-temporal dynamics", *Computer Methods in Applied Mechanics and Engineering*, 2020.

[86] K. T. Carlberg, A. Jameson, M. J. Kochenderfer, J. Morton, L. Peng, and F. D. Witherden, "Recovering missing cfd data for high-order discretizations using deep neural networks and dynamics learning", *Journal of Computational Physics*, 2019.

[87] F. J. Gonzalez and M. Balajewicz, "Deep convolutional recurrent autoencoders for learning low-dimensional feature dynamics of fluid systems", *arXiv preprint arXiv:1808.01346*, 2018.

[88] M. Kalia, S. L. Brunton, H. G. Meijer, C. Brune, and J. N. Kutz, "Learning normal form autoencoders for data-driven discovery of universal, parameter-dependent governing equations", *arXiv preprint arXiv:2106.05102*, 2021.

[89] S. Džeroski, L. Todorovski, I. Bratko, B. Kompare, and V. Križman, "Equation discovery with ecological applications", *Machine learning methods for ecological applications*, 1999.

[90] J. R. Koza, "Genetic programming as a means for programming computers by natural selection", *Statistics and computing*, 1994.

[91] C. Ferreira, "Gene expression programming: A new adaptive algorithm for solving problems", *arXiv preprint cs/0102027*, 2001.

[92] M. F. Javed, M. N. Amin, M. I. Shah, *et al.*, "Applications of gene expression programming and regression techniques for estimating compressive strength of bagasse ash based concrete", *Crystals*, 2020.

[93] J. F. Miller and S. L. Harding, "Cartesian genetic programming", in *Proceedings of the 10th annual conference companion on Genetic and evolutionary computation*, 2008.

[94] R. Miragaia, F. Fernández, G. Reis, and T. Inácio, "Evolving a multiclassifier system for multi-pitch estimation of piano music and beyond: An application of cartesian genetic programming", *Applied Sciences*, 2021.

[95] S.-M. Udrescu and M. Tegmark, "Ai feynman: A physics-inspired method for symbolic regression", *Science Advances*, 2020.

[96] M. Schmidt and H. Lipson, "Distilling free-form natural laws from experimental data", *science*, 2009.

[97] J. Kubalík, E. Derner, J. Žegklitz, and R. Babuška, "Symbolic regression methods for reinforcement learning", *IEEE Access*, 2021.

[98] C. J. Hinde, N. Chakravorti, and A. A. West, "Multi objective symbolic regression", in *Advances in Computational Intelligence Systems: Contributions Presented at the 16th UK Workshop on Computational Intelligence, September 7–9, 2016, Lancaster, UK*, Springer, 2017.

[99] J. Kubalík, E. Derner, and R. Babuška, "Multi-objective symbolic regression for physics-aware dynamic modeling", *Expert Systems with Applications*, 2021.

[100] D. Bertsimas, J. Pauphilet, and B. Van Parys, "Sparse regression: Scalable algorithms and empirical performance", 2020.

[101] S. L. Brunton, J. L. Proctor, and J. N. Kutz, "Discovering governing equations from data by sparse identification of nonlinear dynamical systems", *Proceedings of the national academy of sciences*, 2016.

[102] E. Kaiser, J. N. Kutz, and S. L. Brunton, "Sparse identification of nonlinear dynamics for model predictive control in the low-data limit", *Proceedings of the Royal Society A*, 2018.

[103] W. Heemels, D Lehmann, J Lunze, and B De Schutter, "Introduction to hybrid systems", *Handbook of Hybrid Systems Control–Theory, Tools, Applications*, 2009.

[104] S. Engell, G. Frehse, and E. Schnieder, *Modelling, analysis and design of hybrid systems*. Springer, 2003.

[105] E. R. Westervelt, J. W. Grizzle, and D. E. Koditschek, "Hybrid zero dynamics of planar biped walkers", *IEEE transactions on automatic control*, 2003.

[106] W.-L. Ma, K. A. Hamed, and A. D. Ames, "First steps towards full model based motion planning and control of quadrupeds: A hybrid zero dynamics approach", in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2019.

[107] A. M. Johnson, S. A. Burden, and D. E. Koditschek, "A hybrid systems model for simple manipulation and self-manipulation systems", *The International Journal of Robotics Research*, 2016.

[108] S. Paoletti, A. L. Juloski, G. Ferrari-Trecate, and R. Vidal, "Identification of hybrid systems a tutorial", *European journal of control*, 2007.

[109] R. Rai and C. K. Sahu, "Driven by data or derived through physics? a review of hybrid physics guided machine learning techniques with cyber-physical system (cps) focus", *IEEE Access*, 2020.

[110] D. L. Ly and H. Lipson, "Learning symbolic representations of hybrid dynamical systems", *The Journal of Machine Learning Research*, 2012.

[111] A. Pandala, R. T. Fawcett, U. Rosolia, A. D. Ames, and K. A. Hamed, "Robust predictive control for quadrupedal locomotion: Learning to close the gap between reduced-and full-order models", *IEEE Robotics and Automation Letters*, 2022.

[112] K. A. Hamed, J. Kim, and A. Pandala, "Quadrupedal locomotion via event-based predictive control and qp-based virtual constraints", *IEEE Robotics and Automation Letters*, 2020.

[113] J. Wu, J. Wang, and Z. You, "An overview of dynamic parameter identification of robots", *Robotics and computer-integrated manufacturing*, 2010.

[114] S. Zimmermann, R. Poranne, and S. Coros, "Go fetch!-dynamic grasps using boston dynamics spot with external robotic arm", in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2021.

[115] K. Chatzilygeroudis and J.-B. Mouret, "Using parameterized black-box priors to scale up model-based policy search for robotics", in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2018.

[116] R. T. Fawcett, K. Afsari, A. D. Ames, and K. A. Hamed, "Toward a data-driven template model for quadrupedal locomotion", *IEEE Robotics and Automation Letters*, 2022.