2022 American Control Conference (ACC)
Atlanta, USA, June 8-10, 2022

Feature Learning for Optimal Control with B-spline Representations

Vinay Kenny, Sixiong You, Chaoying Pei, and Ran Dai

***Abstract*— The paper develops a feature learning-based method to solve optimal control problems using B-splines to approximate the optimal solutions. The feature learning-based optimal control method can quickly generate near-optimal trajectories for general optimal control problems subject to system dynamics and constraints. The steps in the proposed method are as follows: Firstly, by representing the state and control variables with B-spline functions, the optimal control problem is converted into an approximate nonlinear programming (NLP) problem, where parameters of the B-splines are identified as features of the optimal solution. Secondly, for a specific problem with designated inputs, a dataset of the optimal trajectories under varying inputs is generated by solving the corresponding NLP problem offline. Finally, the neural network is applied to map the relationship between the designated inputs and identified features, represented by the parameters of B-splines and time variables. To show the effectiveness and efficiency of the proposed method for solving the optimal control problems, extensive simulation cases are presented and analyzed.**

***Index Terms*— Supervised Learning; Optimal Control; B-spline;**

I. INTRODUCTION

Optimal control methods deal with the problem of determining the control policy for a dynamic system that can optimize a specified performance index while satisfying certain constraints [1]. The approaches developed to solve optimal control problems have been classified into two categories, indirect methods and direct methods. Indirect methods are based on the first-order optimality conditions derived from Pontryagin's minimum principle [2]. However, the analytical solution obtained from the indirect methods is available only in very few special cases due to the complexity and nonlinearity in a practical dynamic system. For most of the numerical algorithms combined with the indirect methods, the convergence can not be guaranteed due to the sensitivity of adjoint variables [3].

On the other hand, in a direct method, the constrained optimal control problem is generally transformed into a finite-dimensional nonlinear optimization problem using direct collocation techniques, such as trapezoidal, Runge-Kutta, and Chebyshev [4], [5], [6]. These methods can be implemented to find a local optimal solution but often require a good initial guess [7]. When applied to online calculation, however, numerical performance, e.g., global convergence, cannot be guaranteed for direct methods. In addition, considering the nonconvexity of the reformulated nonlinear optimization problem, many relaxation methods have been developed to further improve the computational efficiency [8]. Even though relaxation methods are time efficient, the discretized optimal trajectories are not continuous or smooth, and require further processing, e.g., interpolation, for onboard implementation. Therefore, improvement is required for both the efficiency and effectiveness of the direct methods while ensuring the smoothness of the optimal trajectory.

To solve optimal control problems in real time, the machine learning method, which is a branch of artificial intelligence, has been studied [9], [10], [11]. For example, [12], [13] applied supervised learning to learn the cost functions, dynamic models, or complex environments to generate onboard feedback control laws. In addition, by mapping the relationship between the initial conditions and the optimal state–control pairs using the supervised learning, a near-optimal solution has been reconstructed in real-time [14]. In [15], without the knowledge about the Hamiltonian parameters, unsupervised learning has been applied to learn a parametric function according to the symplectic gradient of the Hamiltonian function. Then, the well-trained model is able to predict the accurate conserved quantities from data in an unsupervised manner. In addition, via updating the weighting factors of the neural network onboard, reinforcement learning has also been widely applied to solve the optimal control problems [16].

When applying supervised learning to solve optimal control problems, existing approaches construct neural networks to map the relationship between problem inputs and state-control pairs, which requires generating a large-scale dataset and heavy load of computation for data training [14]. To reduce the computational load, our work in [17] finds parameters to represent the features of an optimal control solution when solving bang-bang optimal control problems. However, for general optimal control problems with nonlinear dynamics and constraints, it is challenging to develop a uniform approach to identify features of an optimal solution such that the learning space can be significantly reduced. This paper focuses on developing a uniform approach of identifying features of general optimal control problems by using B-splines to represent the optimal solutions.

The B-spline function has a wide range of applications across different domains, due to its ability to create complex shapes and surfaces using a few parameters. Graphically, a B-spline is a combination of flexible bands that are determined by a number of points called knot points. The key properties of B-spline are local propagation and the ability to select the degree of spline independent of number of control points, which will be discussed in Section III-A. Due to the compact form of B-spline in representing different shapes and the associated properties, it is introduced to represent an optimal solution. Then, parameters of B-splines are identified as features of an optimal solution. A A higher degree B-spline results in increased accuracy of the solution.

To solve the optimal control problem in real time, this paper proposes a B-spline based feature learning (BFL) approach. The advantages of the proposed method include: (1) The proposed BFL approach can be applied to solve general optimal control problems. (2) The neural network needs to train only a few B-spline parameters and time variables, which greatly reduces the complexity of the network and the training time. (3) Successful training of the neural network can quickly generate near-optimal trajectories, which can be used for onboard applications.

The paper's organization is as follows: In Section II, the general formulation of the optimal control problem is introduced. In Section III, the proposed BFL approach is presented in detail. In Section IV, the results of the proposed method in solving a classical optimal control problem are shown and analyzed. Conclusions and future work are presented in Section V.

Vinay Kenny, Sixiong You, Chaoying Pei, and Ran Dai are with the School of Aeronautic and Astronautics, Purdue University, West Lafayette, IN 47907, USA. Emails: vkenny@purdue.edu, you84@purdue.edu, peic@purdue.edu and randai@purdue.edu

978-1-6654-5196-3/$31.00 ©2022 AACC

II. OPTIMAL CONTROL PROBLEM

This section introduces the general optimal control problem of the continuous-time system. Here, we will classify the general optimal control problem into three categories: the unconstrained optimal control problem, optimal control problem with state constraints, and optimal control problem with control constraints. The proposed method considers the following assumption:

***Assumption** 2.1:* The control can be explicitly written in the form of states and/or their derivatives.

A. Unconstrained optimal control problem

The unconstrained optimal control problem of the continuous-time system can be expressed as

$$\begin{aligned} \min_{\mathbf{u},t_f} \quad & J = \phi(t_f, \mathbf{x}(t_f)) + \int_{t_0}^{t_f} L(t, \mathbf{x}, \mathbf{u}) dt \\ \text{s.t.} \quad & \dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}, t) \\ & \mathbf{x}(t_0) = \mathbf{x}_0,\ \psi(\mathbf{x}(t_f), t_f) = 0, \end{aligned} \tag{1}$$

where $\mathbf{u} \in \mathbb{R}^m$ is the control vector, $\mathbf{x} \in \mathbb{R}^n$ is the state vector, t refers to time, J is the objective function, the function ϕ defines the end point cost, the integral of L over the time is the path cost, $\mathbf{f}$ refers to the system dynamics, $\mathbf{x}_0$ represents the values of the initial state vector, function ψ refers to the terminal state constraint, and t_f refers to the terminal time. The objective of (1) is to find the history of the control variable(s) that will minimize a given cost function J, while satisfying the system dynamics and boundary constraints.

B. Optimal control problem with state constraints

The optimal control problem with state constraints can be expressed as

$$\begin{aligned} \min_{\mathbf{u},t_f} \quad & J = \phi(t_f, \mathbf{x}(t_f)) + \int_{t_0}^{t_f} L(t, \mathbf{x}, \mathbf{u}) dt \\ \text{s.t.} \quad & \dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}, t) \\ & \mathbf{g}_i(t, \mathbf{x}) \leq 0,\ i = 1, 2, \ldots, g_n \\ & \mathbf{x}(t_0) = \mathbf{x}_0,\ \psi(\mathbf{x}(t_f), t_f) = 0, \end{aligned} \tag{2}$$

where $\mathbf{g}_i(t, \mathbf{x}) \leq 0,\ i = 1, 2, \ldots, g_n$, represent the state constraints, and g_n is the number of state constraints. Note that, $\mathbf{g}_i(t, \mathbf{x})$ does not contain the control vector and it can either be a linear or non-linear function.

C. Optimal control problem with control constraints

The optimal control problems with control constraints are expressed as

$$\begin{aligned} \min_{\mathbf{u},t_f} \quad & J = \phi(t_f, \mathbf{x}(t_f)) + \int_{t_0}^{t_f} L(t, \mathbf{x}, \mathbf{u}) dt \\ \text{s.t.} \quad & \dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}, t) \\ & \mathbf{h}_i(t, \mathbf{u}) \leq 0,\ i = 1, 2, \ldots, h_n \\ & \mathbf{x}(t_0) = \mathbf{x}_0,\ \psi(\mathbf{x}(t_f), t_f) = 0, \end{aligned} \tag{3}$$

where $\mathbf{h}_i(t, \mathbf{u}) \leq 0,\ i = 1, 2, \ldots, h_n$, represents the control constraints and h_n is the number of control constraints.

Note that, under Assumption 2.1, an optimal control problem with mixed state and control constraints, i.e., $\mathbf{g}_i(t, \mathbf{x}, \mathbf{u}) \leq 0,\ i = 1, 2, \ldots, g_n$, can be classified as an optimal control problem with state constraints, where the control vector can be explicitly expressed as a function of the state vector, denoted as $\mathbf{u} = \mathbf{O}(t, \mathbf{x}, \dot{\mathbf{x}})$. Then the control constraints can be equivalently expressed as $\mathbf{h}_i(t, \mathbf{u}) = \mathbf{h}_i(t, \mathbf{O}(t, \mathbf{x}, \dot{\mathbf{x}})) \leq 0,\ i = 1, 2, \ldots, h_n$.

III. B-SPLINE BASED FEATURE LEARNING APPROACH

In this section, the optimal control problem is reformulated using B-spline representations. Compared with the traditional direct methods based on discretization and collocation techniques, the reformation based on B-splines can reduce the total number of parameters to be optimized, which implies a reduced number of variables for learning. In addition, the smoothness of the optimal trajectory can be guaranteed via the B-spline representations. After solving the B-spline based parameter optimization problem off-line, the solutions are used to construct the dataset for the artificial neural networks. By representing the optimal solution with only a few parameters of B-spline, the required learning space can be significantly reduced.

A. Uniform B-spline function

In most cases, a smooth optimal trajectory can be represented by spline curves. Two of the most commonly used spline curves are the Bezier curve and B-spline curve. Even though both curves can be used to define complicated shapes and surfaces, there are a few limitations associated with Bezier curves. First, it has global propagation. In other words, changing any control point can lead to the change of the entire shape of the curve, as every point on the curve is defined by all the control points. Second, the degree of the Bezier curve depends on the number of control points, which requires spline curves with a higher degree when a large number of control points are involved to define a curve.

In contrast, B-splines is a continuous union of Bezier curve and offers the advantages such as: (1) Local propagation, which means changing any control point affects only a segment defined by this control point, while the rest of the curve remains the same. (2) Uncoupled relationship between the degree of the spline and the number of control points. Thus it offers more flexibility to choose the degree of spline and the number of control points independently. (3) Capability to express complex curves with only a few parameters while ensuring the smoothness of the trajectories [18]. It offers C^{n-1} continuity based on the degree 'n' of the Bezier curve used to develop the n^{th} degree B-spline. For instance, a cubic B-spline is a union of cubic Bezier curves and C^2 continuous, which offers both slope and curvature continuity. These properties of B-spline over the other conventional curves motivate us to use it to define optimal trajectories.

The expression of a uniform B-spline with degree k and $r+1$ control points is

$$S(t) = \sum_{i=0}^{r} N_{i,k}(t) P_i \ ;\ t_{min} \leq t < t_{max} \tag{4}$$

where P_i represents the control point, t_{min} and t_{max} are the minimum and maximum time. $N_{i,k}(t)$ is a basis function defined by Cox-de Boor recusion formula expressed as

$$N_{i,0}(t) = \begin{cases} 1,\ t_i \leq t < t_{i+1} \\ 0, \text{otherwise} \end{cases}$$
$$N_{i,j}(t) = \frac{t - t_i}{t_{i+j} - t_i} N_{i,j-1}(t) + \frac{t_{i+j+1} - t}{t_{i+j+1} - t_{i+1}} N_{i+1,j-1}(t)$$

where $\mathbf{T} = (t_0, t_1, \ldots t_m)$ is called a knot vector, and its elements are knot points. The total number of knot points for the above formulation is $m+1$, where m is given by $m = k + r + 1$. The knot vector determines which basis function affects the shape of B-spline. The degree 'k' can be selected depending on the accuracy requirements of the solution, where $k \in [2, r+1]$.

A uniform B-splines is a B-spline segmented in equal steps such that the distance between any two adjacent knots is equivalent. For example, the quadratic uniform B-spline for $t \in [0,1]$ has three active basis functions, which can be expressed in a matrix form as

$$b(t) = \frac{1}{2!}\begin{bmatrix} P_0 & P_1 & P_2 \end{bmatrix} \begin{bmatrix} 1 & -2 & 1 \\ -2 & 2 & 1 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} t^2 \\ t \\ 1 \end{bmatrix}. \tag{5}$$

In general the B-spline function of degree k can be expressed as

$$b(t) = \frac{1}{k!}\mathbf{P}^T\mathbf{A}\mathbf{t}, \tag{6}$$

where $\mathbf{P} = [P_0, ..., P_{k-1}, P_k]^T \in \mathbb{R}^{k+1}$ is a parameter vector, $\mathbf{A} \in \mathbb{R}^{(k+1)\times(k+1)}$, and $\mathbf{t} = [t^k, t^{k-1}, ..., t^0]^T \in \mathbb{R}^{k+1}$ are constant matrices. A more detailed description of B-splines can be found in [19].

B. B-spline representation for the optimal control solution

For the unconstrained optimal control problem, by assuming that the optimal state vector $\mathbf{x}$ is smooth, the state vector $\mathbf{x}$ can be approximated by a polynomial function of t. However, once state constraints and control constraints are considered, the state vector $\mathbf{x}$ is not necessarily smooth. In that case, we assume that the state vector $\mathbf{x}$ can be approximated via piece-wise polynomial functions. Then in the following part, we will divide the optimal control problem into three categories, the unconstrained optimal control problem, the optimal control problem with state constraints, and the optimal control problem with control constraints. As we have mentioned that the optimal control problems with mixed state and control constraints can be regarded as the optimal control problem with state constraints, it will not be discussed in the following part. Then, the process of applying the B-spline function to reformulate the optimal control problem will be discussed case by case.

Type 1, Unconstrained optimal control problem: In the unconstrained optimal control problem (1), each state and control variable can be expressed as a B-spline function, denoted as $\mathbf{X} = \mathbf{x}(\mathbf{\Omega}_x, t) = \frac{1}{k!}\mathbf{\Omega}_x\mathbf{A}\mathbf{t}$ and $\mathbf{U} = \mathbf{u}(\mathbf{\Omega}_u, t) = \frac{1}{k!}\mathbf{\Omega}_u\mathbf{A}\mathbf{t}$, where $\mathbf{\Omega}_x = \left[\mathbf{P}_{x_1}, \mathbf{P}_{x_2}, ..., \mathbf{P}_{x_n}\right]^T \in \mathbb{R}^{n\times(r+1)}$ and $\mathbf{\Omega}_u = \left[\mathbf{P}_{u_1}, \mathbf{P}_{u_2}, ..., \mathbf{P}_{u_m}\right]^T \in \mathbb{R}^{m\times(r+1)}$. Thus, the optimal control problem can be reformulated as

$$\begin{aligned} \min_{\mathbf{\Omega}_x, \mathbf{\Omega}_u, t_f} \quad & J = \phi(t_f, \mathbf{X}_{t_f}) + \int_{t_0}^{t_f} L(t, \mathbf{X}, \mathbf{U})dt \\ \text{s.t.} \quad & \mathbf{X} = \mathbf{x}(\mathbf{\Omega}_x, t) = \frac{1}{k!}\mathbf{\Omega}_x\mathbf{A}\mathbf{t} \\ & \mathbf{U} = \mathbf{u}(\mathbf{\Omega}_u, t) = \frac{1}{k!}\mathbf{\Omega}_u\mathbf{A}\mathbf{t} \\ & \dot{\mathbf{X}} = \mathbf{f}(\mathbf{X}, \mathbf{U}, t) \\ & \mathbf{X}_{t_0} = \mathbf{x}_0, \ \psi(\mathbf{X}_{t_f}, t_f) = 0. \end{aligned} \tag{7}$$

Therefore, for unconstrained optimal control problems, the unknown variables, including the B-spline parameters, $\mathbf{\Omega}_x$ and $\mathbf{\Omega}_u$, and the final time t_f, are identified as features for this type of problem.

Type 2, Optimal control problem with state constraints: A state constrained optimal control problem applies constraint on state variables, where the constraint can be a linear or nonlinear function. When the state constraints are active along the optimal trajectory for a nontrivial interval, these constraints force the trajectory to be divided into multiple segments that are not necessarily smooth.

To better illustrate the optimal solutions of the optimal control problems with state constraints, a diagram of the optimal trajectory for an optimal control problem with state constraints is shown in Fig. 1, where the state constraint $\mathbf{g}_i(t, \mathbf{x}) \le 0$ is marked with dashed line, and the optimal solution is represented by the solid line. It can be found that in Fig. 1 the state constraint is active along two intervals, denoted as intervals $[t_{b1}, t_{b2}]$ and $[t_{b3}, t_{b4}]$. Note that, for the nontrivial interval where $t_{b1} \ne t_{b2}$, it is named boundary interval, and for the trivial interval where $t_{b3} = t_{b4}$, it is named contact point. For simplicity but without loss of generality, we will explain how to identify features for an optimal trajectory with multiple segments, where only one boundary interval and one contact point are included, as shown in Fig. 1.

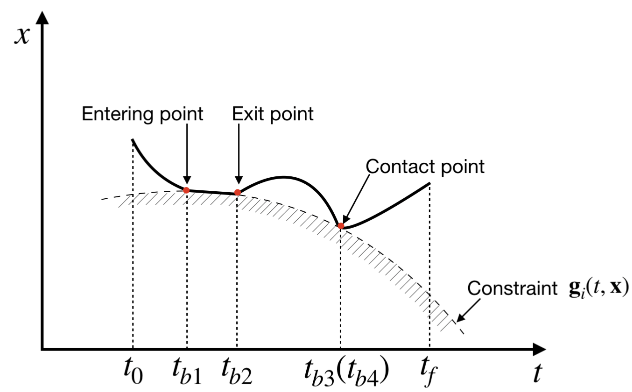

Fig. 1: A diagram of the optimal trajectory for an optimal control problem with state constraints

It can be found that the optimal trajectory in Fig. 1 is divided into 4 segments, $[t_0, t_{b1}]$, $[t_{b1}, t_{b2}]$, $[t_{b2}, t_{b3}]$ and $[t_{b4}, t_f]$. Then, we need 4 independent B-spline functions to represent the features of all segments and the time variables t_{b1}, t_{b2}, t_{b3}, and t_f to represent the remaining parts of features for the optimal solution. In fact, the contact point with an active state constraint can be regarded as a special case of the boundary interval with $t_{b3} = t_{b4}$. Therefore, to make the BFL approach general, a B-spline function is also assigned to the interval $[t_{b3}, t_{b4}]$ for active state constraint on the contact point.

According to the above analysis, for a finite interval $[t_0, t_f]$, there are a finite number of boundary intervals and contact points. When the number of boundary intervals and contact points are determined, denoted by K, parameters of $2K+1$ independent B-spline functions and $2K+1$ time parameters are identified as features for the optimal solution with state constraints. Thus, the optimal control problem can be reformulated as

$$\begin{aligned} \min_{\mathbf{\Omega}_x^j, \mathbf{\Omega}_u^j, t_{b,j}} \quad & J = \phi(t_f, \mathbf{X}_{t_f}) + \sum_{j=2}^{2K} \int_{t_{b,j-1}}^{t_{b,j}} L(t, \mathbf{X}^j, \mathbf{U}^j)dt \\ & + \int_{t_0}^{t_{b,1}} L(t, \mathbf{X}^1, \mathbf{U}^1)dt + \int_{t_{b,2K}}^{t_f} L(t, \mathbf{X}^{2K+1}, \mathbf{U}^{2K+1})dt \\ \text{s.t.} \quad & \mathbf{X}^j = \mathbf{x}(\mathbf{\Omega}_x^j, t) = \frac{1}{k!}\mathbf{\Omega_x}^j\mathbf{A}\mathbf{t}, \ j = 1, 2, ..., 2K+1 \\ & \mathbf{U}^j = \mathbf{u}(\mathbf{\Omega}_u^j, t) = \frac{1}{k!}\mathbf{\Omega_u}^j\mathbf{A}\mathbf{t}, \ j = 1, 2, ..., 2K+1 \\ & \dot{\mathbf{X}}^j = \mathbf{f}(\mathbf{X}^j, \mathbf{U}^j, t), \ j = 1, 2, ..., 2K+1 \\ & \mathbf{g}_i(t, \mathbf{X}^j) \le 0, \ j = 1, 2, ..., 2K+1, \ i = 1, 2, \ldots, g_n \\ & \mathbf{g}_i(t, \mathbf{x}(\mathbf{\Omega}_x^j, t)) = 0, \ t_{b,j} \le t \le t_{b,j+1}, \\ & j = 1, 3, ..., 2K-1, \ i = 1, 2, \ldots, g_n \\ & \mathbf{X}_{t_0} = \mathbf{x}_0, \ \psi(\mathbf{X}_{t_f}, t_f) = 0, \end{aligned} \tag{8}$$

where $t_{b,j}$ and $t_{b,j+1}$ are the starting and ending time of the jth

B-spline segment for $j = 1, 3, ..., 2K-1$.

However, the number of boundary intervals and contact points are not directly given before solving problem (8). The number K can be determined by either the optimal control theory or numerical search. For instance, for the optimal control problem (2) with $m = n = 1$, the third-order state constraint will only have one contact point according to the Pontryagin's maximum principle [20]. Thus, we can determine $K = 1$ for that special case. However, for the other cases with $m > 1$ or $n > 1$, the maximum number of boundary intervals and contact points cannot be determined analytically. For those cases, numerical search will be applied to gradually increase the integer number K within a range till the minimum value of problem (8) is obtained.

Type 3, Optimal control problem with control constraints: Similar to the state constrained problem discussed above, a control constrained optimal control problem can also be processed in the same way. When the control constraints are active along the optimal control profile, the optimal control and trajectory are divided into multiple segments. Assuming that the number of the boundary intervals and the contact points is finite in the interval $[t_0, t_f]$, and G refers to the number of boundary intervals and contact points. Thus, in such a problem, parameters of $2G+1$ independent B-spline functions and $2G+1$ time parameters are the features for the coreresponding problem. The optimal control problem can be rewritten as

$$\min_{\mathbf{\Omega}_x^j, \mathbf{\Omega}_u^j, t_{b,j}} J = \phi(t_f, \mathbf{X}_{t_f}) + \sum_{j=2}^{2G} \int_{t_{b,j-1}}^{t_{b,j}} L(t, \mathbf{X}^j, \mathbf{U}^j) dt \quad (9)$$
$$+ \int_{t_0}^{t_{b,1}} L(t, \mathbf{X}^1, \mathbf{U}^1) dt + \int_{t_{b,2G}}^{t_f} L(t, \mathbf{X}^{2G+1}, \mathbf{U}^{2G+1}) dt$$
$$\text{s.t.} \quad \mathbf{X}^j = \mathbf{x}(\mathbf{\Omega}_x^j, t) = \frac{1}{k!} \mathbf{\Omega}_x^j \mathbf{A} \mathbf{t}, \ j = 1, 2, ..., 2G+1$$
$$\mathbf{U}^j = \mathbf{u}(\mathbf{\Omega}_u^j, t) = \frac{1}{k!} \mathbf{\Omega}_u^j \mathbf{A} \mathbf{t}, \ j = 1, 2, ..., 2G+1$$
$$\dot{\mathbf{X}}^j = \mathbf{f}(\mathbf{X}^j, \mathbf{U}^j, t), \ j = 1, 2, ..., 2G+1$$
$$\mathbf{h}_i(t, \mathbf{U}^j) \le 0, \ j = 1, 2, ..., 2G+1, i = 1, 2, \ldots, h_n$$
$$\mathbf{h}_i(t, \mathbf{u}(\mathbf{\Omega}_u^j, t)) = 0, \ t_{b,j} \le t \le t_{b,j+1},$$
$$j = 1, 3, ..., 2G-1, \ i = 1, 2, \ldots, h_n$$
$$\mathbf{X}_{t_0} = \mathbf{x}_0, \ \psi(\mathbf{X}_{t_f}, t_f) = 0.$$

C. Offline solution for optimal control problems with B-spline representations

The B-spline representation for the optimal control problems discussed above identify features for the three types of optimal control problem. However, we cannot solve the formulated problems in (7)-(9) directly to find the values of the identified features. Since the state and control are represented by B-spline functions in the system dynamics, denoted by $\dot{\mathbf{X}} = \mathbf{f}(\mathbf{X}, \mathbf{U}, t)$, the dynamics cannot be satisfied all the time, over the time interval $[t_0, t_f]$ for a limited number of B-spline parameters with a given degree. To approximately satisfy the system dynamics, discrete knot points are applied to the B-spline function, which converts the parameter optimization problems in (7) - (9) into NLP problems. Then, an NLP problem needs to be solved to determine values of identified features under given inputs, e.g., boundary conditions and/or parameters in the dynamics. For simplicity, the conversion of the unconstrained optimal control problem (1) into an NLP is illustrated here. With the whole time interval $[t_0, t_f]$ being equally discretized into $[t_0, t_1, t_2, ..., t_Q]$, where $t_0 < t_1 < t_2 < ... < t_Q = t_f$, the unconstrained optimal control problem with a single state and control can be reformulated as

$$\min_{\mathbf{\Omega}_x, \mathbf{\Omega}_u, t_f} \quad J = \phi(t_f, \mathbf{X}_{t_f}) + \sum_{q=1}^{Q} L(t_q, \mathbf{X}_q, \mathbf{U}_q) \quad (10)$$
$$\text{s.t.} \quad \mathbf{X}_q = \mathbf{x}(\mathbf{\Omega}_x, t_q) = \frac{1}{k!} \mathbf{\Omega}_x \mathbf{A} \mathbf{t}_q, \ q = 1, 2, ...Q$$
$$\mathbf{U}_q = \mathbf{x}(\mathbf{\Omega}_u, t_q) = \frac{1}{k!} \mathbf{\Omega}_u \mathbf{A} \mathbf{t}_q, \ q = 1, 2, ...Q$$
$$\dot{\mathbf{X}}_q = \mathbf{f}(\mathbf{X}_q, \mathbf{U}_q, t_q), \ q = 1, 2, ...Q$$
$$\mathbf{X}_0 = \mathbf{x}_0, \ \psi(\mathbf{X}_{t_f}, t_f) = 0.$$

By solving the reformulated problem (10), the values of identified features, $\mathbf{\Omega}_x$, $\mathbf{\Omega}_u$, and t_f, can be determined. Similarly, the values of features identified in (8) and (9) can be obtained by applying discretization to each B-spline segment and then solving the reformulated NLP problem.

D. B-spline based Feature Learning Approach

With the B-spline representation for the optimal control solution, the remaining part is to apply the neural network to learn these identified features. Figure 2 shows the structure of the neural network for the optimal control problem with identified features. The neural network is comprised of three distinct types of layers, input layer, hidden layer, and output layer. In each layer, multiple activation functions are included. Depending on the trends being observed in outputs, a proper combination of linear and nonlinear activation function is required to achieve fast convergence rate along with minimum error between the learnt data and the actual data. More information on the artificial neural network can be found in [21].

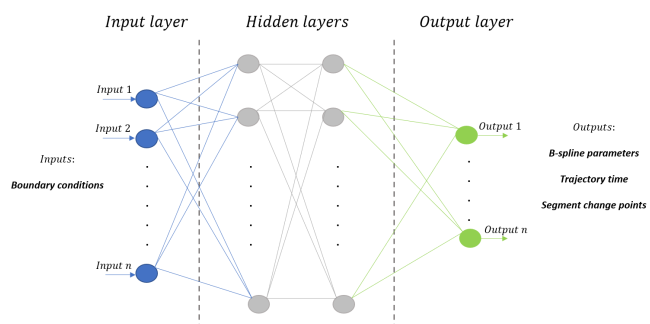

Fig. 2: Structure of multilayered feedforward neural network

The implementation of the BFL approach contains two main parts: offline and online, as described by the flowchart in Table I. For the offline part, there are four steps. First, the original optimal control problem is translated into an approximate B-spline based parameter optimization problem subject to dynamics, boundary constraints, and state/control constraints, where each state/control is represented by a B-spline function. Second, for a studied optimal control problem, the ranges of inputs, e.g., boundary conditions and/or parameters in the dynamics, are determined. Then, for each case with given inputs, the corresponding B-spline and time parameters, identified as features of the optimal control solution are obtained via solving the formulated NLP problem (10). By solving a sufficient number of cases within the selected input range, a sufficiently large dataset is generated, where the inputs are boundary conditions and/or dynamics parameters and outputs are B-spline and time parameters. Finally, the neural networks can be constructed and trained to map the relationship between inputs and outputs of

TABLE I: Flowchart of BFL

Algorithm: B-spline based feature learning approach
Off-line part:
1) Determine the type of the optimal control problem and reformulated it into problem (9);
2) Select the range of inputs for (10);
3) Via solving (10) within the selected range, a dataset of optimized $\mathbf{\Omega}_x$, $\mathbf{\Omega}_u$, and time parameters under varying inputs is generated;
4) Construct and train the neural networks to map the relationship between inputs and $\mathbf{\Omega}_x$, $\mathbf{\Omega}_u$, and time parameters;
On-line part:
1) Calculate $\mathbf{\Omega}_x$, $\mathbf{\Omega}_u$, and time parameters according to the trained neural networks under given inputs
2) Reconstruct the optimal control and state via $\mathbf{\Omega}_x$, $\mathbf{\Omega}_u$, and time parameters

the generated dataset. For the on-line part, there are two steps. First, for a given input set, the optimized parameters $\mathbf{\Omega}_x$, $\mathbf{\Omega}_u$, and t_f can be found via the trained neural network. Next, the optimal control and state variables can be reconstructed on-board via $\mathbf{x}(\mathbf{\Omega}_x, t) = \frac{1}{k!}\mathbf{\Omega}_x\mathbf{A}\mathbf{t}$ and $\mathbf{u}(\mathbf{\Omega}_u, t) = \frac{1}{k!}\mathbf{\Omega}_u\mathbf{A}\mathbf{t}$.

IV. SIMULATION RESULTS

To verify the effectiveness of the proposed method, a classical optimal control problem, the Brachistochrone problem is presented in this section. The results obtained from the proposed BFL method are analyzed and compared to the analytical and/or NLP solutions. All the simulations are conducted on a laptop with i7-8750H CPU @ 2.20GHz and 16GB of RAM.

The unconstrained Brachistochrone problem can be formulated as:

$$\begin{gathered} \min_{u,t_f} J = \int_{t_0}^{t_f} 1dt \\ \text{subject to } \dot{x} = Vcos(u),\ \dot{y} = Vsin(u) \\ V = \sqrt{2gy} \\ x(t_0) = x_0, y(t_0) = y_0, x(t_f) = x_f, y(t_f) = y_f, \end{gathered} \tag{11}$$

where V is the velocity of the particle and u is the angle between the direction of V and the x-axis, $[x, y]$ represents the position of the particle, t_0 and t_f are the starting and ending time, respectively. To satisfy the conservation of energy, $V = \sqrt{2gy}$. The objective is to minimize t_f, while satisfying the dynamics, boundary constraints, and the energy conservation constraint.

In general, the higher the degree of B-spline is, the higher the accuracy of the generated trajectory will be. To balance between the computational load and accuracy, B-splines with degree $k = 4$ is used to represent the states in this problem. The constant matrix $\mathbf{A}$ in (6) for the fourth degree B-spline with $t \in [0, 1)$ and 5 control points is given by:

$$\begin{bmatrix} 1 & -4 & 6 & -4 & 1 \\ -4 & 12 & -6 & -12 & 11 \\ 6 & -12 & -6 & 12 & 11 \\ -4 & 4 & 6 & 4 & 1 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

Then the states vector can be written as $\mathbf{X} = [x, y]$ and expressed as

$$\mathbf{X}(\tau) = \frac{1}{4!}\mathbf{\Omega}_x\mathbf{A}\mathbf{T}, \tag{12}$$

where $\mathbf{\Omega}_x = [\mathbf{P}_1, \mathbf{P}_2]^T \in \mathbb{R}^{2\times 5}$ represents the parameter matrix for the two B-splines and $\mathbf{P}_1, \mathbf{P}_2 \in \mathbb{R}^5$ are two parameter vectors associated with x and y, respectively. In addition, in equation (12), $\mathbf{T} = [\tau^4, \tau^3, \tau^2, \tau, 1]$ and $\tau \in [0, 1]$ is the unified time and the relationship between τ and the real time t can be expressed as

$$t = (t_f - t_0)\tau + t_0. \tag{13}$$

With $\mathbf{\Omega}_x$ and A being constant matrices, $\dot{\mathbf{X}}$ can be expressed as

$$\dot{\mathbf{X}}(\tau) = \frac{1}{4!}\mathbf{\Omega}_x\mathbf{A}\dot{\mathbf{T}}, \tag{14}$$

where $\dot{\mathbf{T}}$ can be expressed as $[\frac{4\tau^3}{t_f}, \frac{3\tau^2}{t_f}, \frac{2\tau}{t_f}, \frac{1}{t_f}, 0]^T$. For this problem, since we have the boundary conditions

$$\begin{gathered} \mathbf{X}(0) = [x_0, y_0]^T, \\ \mathbf{X}(1) = [x_f, y_f]^T \end{gathered} \tag{15}$$

we can further reduce the number of B-spline parameters. Here we denote $\mathbf{P}_1 = [P_{1,0}, P_{1,1}, P_{1,2}, P_{1,3}, P_{1,4}]$, $\mathbf{P}_2 = [P_{2,0}, P_{2,1}, P_{2,2}, P_{2,3}, P_{2,4}]$, by substituting equation (15) into (14), the parameters P_{10}, P_{14}, P_{20} and P_{24} can be obtained by

$$\begin{aligned} P_{1,0} &= 24x_0 - 11P_{1,1} - 11P_{1,2} - P_{1,3}, \\ P_{1,4} &= 24x_f - P_{1,1} - 11P_{1,2} - 11P_{1,3}, \\ P_{2,0} &= 24y_0 - 11P_{2,1} - 11P_{2,2} - P_{2,3}, \\ P_{2,4} &= 24y_f - P_{2,1} - 11P_{2,2} - 11P_{2,3}. \end{aligned} \tag{16}$$

Thus, in the vector $\mathbf{P}_1$ and $\mathbf{P}_2$, only 6 variables need to be determined, which are $P_{1,1}, P_{1,2}, P_{1,3}$ and $P_{2,1}, P_{2,2}, P_{2,3}$.

According to (11), the variable u can be explicitly expressed by states for this specific problem, written as

$$\begin{gathered} \dot{x}^2 + \dot{y}^2 = V^2, \\ V = \sqrt{2gy}. \end{gathered} \tag{17}$$

Therefore, the B-spline function for representation of the control variable is not required for this problem. By evenly discretizing $\tau \in [0, 1]$ into 7 points $[\tau_1, \tau_2, ...\tau_7]$, problem (11) can be rewritten as an NLP problem,

$$\begin{aligned} \min_{\mathbf{P}_1,\mathbf{P}_2,t_f} \quad & J = t_f, \\ \text{subject to} \quad & \dot{x}_q^2 + \dot{y}_q^2 = {V_q}^2,\ q = 1, 2, ...5, \\ & x_q = \frac{1}{4!}\mathbf{P}_1^T\mathbf{A}\mathbf{T}_q,\ q = 1, 2, ...5, \\ & y_q = \frac{1}{4!}\mathbf{P}_2^T\mathbf{A}\mathbf{T}_q,\ q = 1, 2, ...5, \\ & V_q = \sqrt{2gy_q},\ q = 1, 2, ...5, \\ & x_1 = x_0,\ y_1 = y_0, \\ & y_7 = x_f,\ y_7 = y_f, \end{aligned} \tag{18}$$

where $\mathbf{T}_q = [\tau_q^4, \tau_q^3, \tau_q^2, \tau_q, 1]^T$, x_q, y_q, and V_q represent the values of x, y and V at the qth collocation point, respectively.

The proposed BFL approach is applied to solve the Brachistochrone problem with different boundary conditions. Different combinations of boundary conditions can be used to evaluate the computational performance and effectiveness of the BFL approach. For the simulation cases shown in this section, the boundary conditions for generating the dataset are selected to be $x_0 = 0$, $y_0 = 0$, $x_f = [4, 7]$ and $y_f = -10$, and the size of the dataset is chosen to be 1000. By solving problem (18) with a varying x_f in the range of $[4, 7]$, the dataset is generated, where input and output are x_f and $[\mathbf{P}_1, \mathbf{P}_2, t_f]$, respectively.

In addition, from the generated dataset, the data points for $x_f \in [4, 6.69]$ (900 data points) were used for training the neural network, including 80% for training data, 10% for test data, and 10% for validation data. Therefore, $x_f \in (6.69, 7]$ are points outside the training dataset and can be used for validate the effectiveness of the BFL approach outside the training dataset. Then, the neural network is applied to map the relationship between input and output. In this example, a neural network with four layers is constructed and the number of neurons in each layer is 20, 20, 20, and 10, respectively. Additionally, the activation functions include two sigmoid functions, one hyperbolic function, and one ReLu function. Then the simulation results of the Brachistochrone problem with different constraints are presented separately.

A. Case 1: Unconstrained Brachistochrone problem

To verify the effectiveness of the BFL approach, the case with $x_f = 7$, which is outside the generated dataset, is simulated and shown in Figure 3. Additionally, the optimal trajectory from analytical solution, optimized trajectory via solving the problem in (18), and the trajectory from BFL are marked with green solid line, black dash line, and red points, respectively. It can be found that all three trajectories are close to each other, which means the trajectory from BFL is near optimal. The computational time of NLP solver and BFL approach are 0.613 seconds and 0.018 seconds, respectively. It indicates the real-time performance of the proposed BFL approach compared to the direct methods. In addition, 1000 cases are simulated and compared by randomly selecting $x_f \in [4, 7]$. Comparing solutions of BFL with the optimal ones, the root mean square error (RMSE) of t_f is found to be 0.0037962 seconds. Thus, we can conclude that the BFL approach is effective for solving the unconstrained Brachistochrone problem.

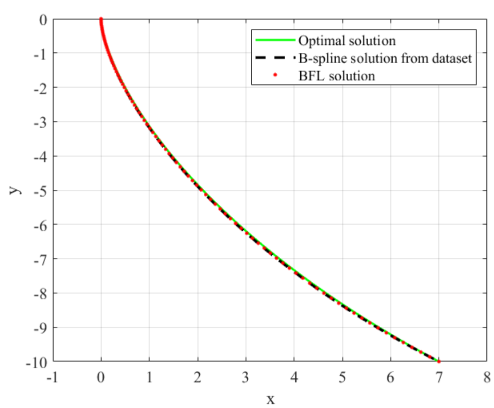

Fig. 3: Unconstrained trajectories for $x_0 = 0$, $y_0 = 0$, $x_f = 7$ and $y_f = -10$.

B. Case 2: Brachistochrone problem with state constraints

In the state constrained Brachistochone problem, the state constraint is considered, expressed as

$$y \geq -2.5x - 0.25. \tag{19}$$

Figure 4 shows the relationship between the additional linear constraint and the unconstrained solutions obtained in Section IV-A, where the blue and orange curves are the solutions of the unconstrained problem with $x_f = 4$ and $x_f = 7$, respectively, while the pink curve represents the additional linear constraint in (19). It is obvious that the original unconstrained Brachistochrone problem cannot satisfy the state constraint specified above.

The simulation result in Figure 5 shows that the optimal trajectory with the state constraint is composed of 3 segments, where the first and third segments can be expressed with B-spline functions and the second segment overlaps with the linear state constraint. To determine such a trajectory, besides the parameters for B-spline functions, the terminal time for each segment are also needed to be determined.

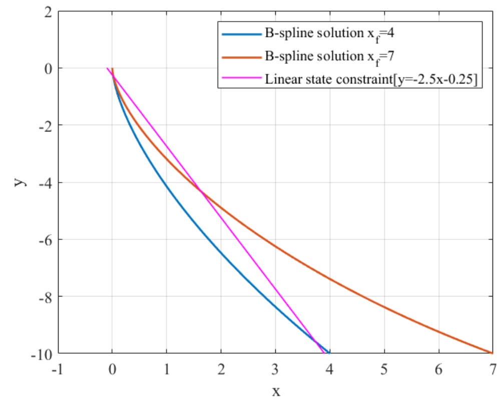

Fig. 4: Unconstrained B-spline trajectories for $x_f = 4$ and $x_f = 7$ and the constraint line $y = -2.5x - 0.25$

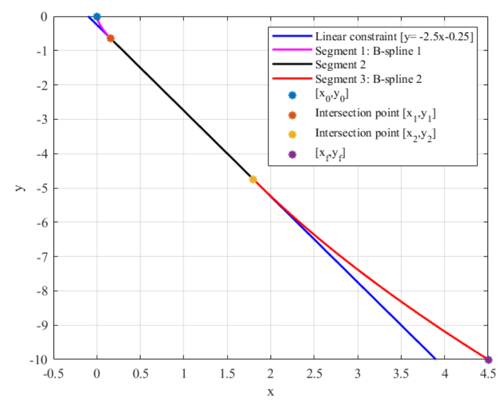

Fig. 5: Trajectory segments for $x_f = 4.5$ and the state constraint $y = -2.5x - 0.25$

Figure 6 shows the simulation results of the Brachistochrone problem with the state constraint and terminal $x_f = 7$, where the green solid curve shows the optimal trajectory obtained by solving the problem directly from collocation and NLP, the black dashed curve represents the B-spline trajectory obtained by solving (18), and the red curve denotes the trajectory from BFL. The plots verify that the trajectory from BFL is almost identical to the NLP solution. Moreover, all the points on the BFL trajectory satisfy the linear state constraint. The computational time of NLP solver and BFL approach are 0.932 seconds and 0.019 seconds, respectively, which indicates the real-time performance of the proposed BFL approach again in solving optimal control problems with state constraints. 1000 cases are simulated with random x_f selected within the range $x_f \in [4, 7]$. The RMSE of t_f between the BFL solution and the NLP solution is found to be 0.0199 seconds. Thus, the BFL method can effectively solve the state constrained Brachistochrone problem.

C. Case 3: Brachistochrone problem with control constraints

The control constraints that added to the Brachistochrone problem is stated as

$$\dot{u} \leq 1 \; rad/s, \tag{20}$$

which indicates that the changing rate of path angle u is upper bounded. Combining with the dynamics of the Brachistochrone problem, (20) can be further expressed as

$$\dot{u} = \frac{\ddot{y}\dot{x} - \dot{y}\ddot{x}}{\dot{x}^2 + \dot{y}^2} \leq 1. \tag{21}$$

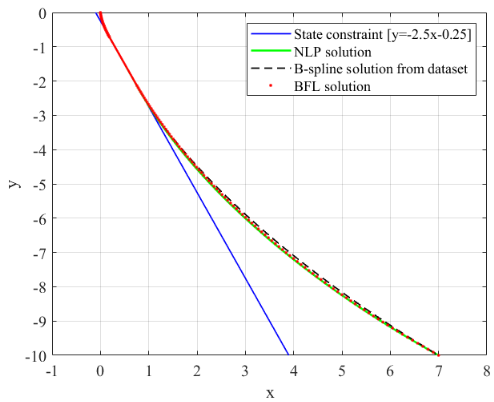

Fig. 6: State constrained trajectories for $x_f = 7$ and the constraint line $y = -2.5x - 0.25$

The simulation results of the Brachistochrone problem with the control constraint for $x_f = 7$ are shown in Figure 7, where the optimized trajectory from NLP solver, B-spline trajectory via solving (18), and trajectory from BFL are presented. It can be found that all trajectories are close to each other, which verifies that the trajectory from BFL is near optimal. The computational time of NLP solver and BFL approach are 0.843 seconds and 0.018 seconds, respectively. It again indicates the real-time performance of the proposed BFL approach in solving optimal control problems with control constraints. In addition, 1000 cases are simulated by randomly selecting $x_f \in [4, 7]$. Comparing solutions of BFL with the solutions from NLP, the RMSE of t_f is found to be 0.0031 seconds. Thus, the BFL approach is effective in solving the control constrained Brachistochrone problem.

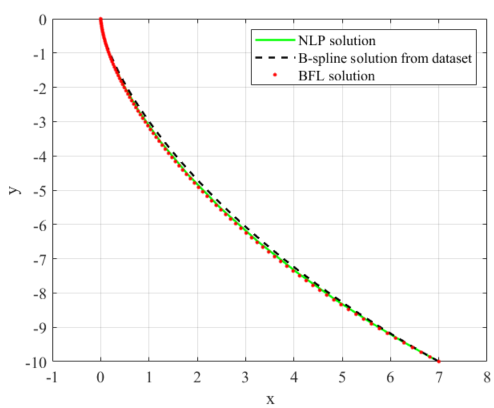

Fig. 7: Control constrained B-spline trajectories for x_f= 7

V. CONCLUSIONS

This paper develops a B-spline based feature learning approach to solve the optimal control problem in real-time. First, the optimal control problem is reformulated into an approximate parameter optimization problem, and the optimal solution is represented with only a few B-spline and time parameters, identified as features of the optimal solution. Then, a dataset of optimized B-spline parameters is generated for training via solving the reformulated parameter optimization problem with varying inputs. By representing the optimal solutions with only a few identified parameters, greatly reduces the complexity of the neural network and computational cost for training. In addition, from the simulation results of the Brachistochrone problem, it can be found that the computational time of the proposed approach is less than 0.02 seconds for all cases. At the same time a near optimal solution can be reconstructed. Thus, we can conclude that the proposed B-spline based feature learning approach can be promising in finding a near optimal solution along with real-time computational performance.

REFERENCES

[1] A. E. Bryson, Y.-C. Ho, and G. M. Siouris, "Applied optimal control: Optimization, estimation, and control," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 9, no. 6, pp. 366–367, 1979.

[2] T. J. Böhme and B. Frank, *Indirect Methods for Optimal Control*. Cham: Springer International Publishing, 2017, pp. 215–231. [Online]. Available: https://doi.org/10.1007/978-3-319-51317-1_7

[3] S. You, R. Dai, and P. Lu, "Learning-based hamilton-jacobi-bellman methods for optimal control," *arXiv preprint arXiv:1907.10097*, 2019.

[4] W. Zhang and H. Ma, "The chebyshev–legendre collocation method for a class of optimal control problems," *International Journal of Computer Mathematics*, vol. 85, no. 2, pp. 225–240, 2008.

[5] F. Fahroo and I. M. Ross, "Direct trajectory optimization by a chebyshev pseudospectral method," *Journal of Guidance, Control, and Dynamics*, vol. 25, no. 1, pp. 160–166, 2002.

[6] G. T. Huntington and A. V. Rao, "A comparison between global and local orthogonal collocation methods for solving optimal control problems," in *2007 American Control Conference*, 2007, pp. 1950–1957.

[7] D. R. Jones and J. R. Martins, "The direct algorithm: 25 years later," *Journal of Global Optimization*, vol. 79, no. 3, pp. 521–566, 2021.

[8] X. Liu and P. Lu, "Solving nonconvex optimal control problems by convex optimization," *Journal of Guidance, Control, and Dynamics*, vol. 37, no. 3, pp. 750–765, 2014.

[9] P. V. Medagam and F. Pourboghrat, "Optimal control of nonlinear systems using rbf neural network and adaptive extended kalman filter," in *2009 American Control Conference*. IEEE, 2009, pp. 355–360.

[10] K. S. B. Z. I. D. and P. P. E., "Machine learning: a review of classification and combining techniques," *Artificial Intelligence Review*, vol. 26, no. 3, pp. 159–190, Nov 2006. [Online]. Available: https://doi.org/10.1007/s10462-007-9052-3

[11] O. Y. Al-Jarrah, P. D. Yoo, S. Muhaidat, G. K. Karagiannidis, and K. Taha, "Efficient machine learning for big data: A review," *Big Data Research*, vol. 2, no. 3, pp. 87 – 93, 2015, big Data, Analytics, and High-Performance Computing. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2214579615000271

[12] J. W. Kim, B. J. Park, H. Yoo, T. H. Oh, J. H. Lee, and J. M. Lee, "A model-based deep reinforcement learning method applied to finite-horizon optimal control of nonlinear control-affine system," *Journal of Process Control*, vol. 87, pp. 166–178, 2020.

[13] J. Park, Z. Chen, L. Kiliaris, M. L. Kuang, M. A. Masrur, A. M. Phillips, and Y. L. Murphey, "Intelligent vehicle power control based on machine learning of optimal control parameters and prediction of road type and traffic congestion," *IEEE Transactions on Vehicular Technology*, vol. 58, no. 9, pp. 4741–4756, 2009.

[14] C. Sánchez-Sánchez and D. Izzo, "Real-time optimal control via deep neural networks: study on landing problems," *Journal of Guidance, Control, and Dynamics*, vol. 41, no. 5, pp. 1122–1135, 2018.

[15] S. J. Greydanus, M. Dzumba, and J. Yosinski, "Hamiltonian neural networks," 2019.

[16] D. Bertsekas, *Reinforcement and Optimal Control*. Athena Scientific, 2019.

[17] S. You, C. Wan, R. Dai, and J. R. Rea, "Learning-based onboard guidance for fuel-optimal powered descent," *Journal of Guidance, Control, and Dynamics*, vol. 44, no. 3, pp. 601–613, 2021.

[18] R. Dai, "B-splines based optimal control solution," in *AIAA Guidance, Navigation, and Control Conference*, 2010, p. 7888.

[19] C. De Boor and C. De Boor, *A practical guide to splines*. springer-verlag New York, 1978, vol. 27.

[20] W. Hamilton, "On nonexistence of boundary arcs in control problems with bounded state variables," *IEEE Transactions on Automatic Control*, vol. 17, no. 3, pp. 338–343, 1972.

[21] A. Abraham, "Artificial neural networks," *Handbook of measuring system design*, 2005.