Research Article

A Simple and Accurate Apollo-Trained Neural Network Controller for Mars Atmospheric Entry

Hao Wang and Tarek A. Elgohary

Department of Mechanical and Aerospace Engineering, University of Central Florida, Orlando, FL 32816, USA

Correspondence should be addressed to Hao Wang; wanghao@knights.ucf.edu

Received 22 January 2020; Revised 8 May 2020; Accepted 26 July 2020; Published 9 September 2020

Academic Editor: James D. Biggs

Copyright © 2020 Hao Wang and Tarek A. Elgohary. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

We present a new method to design the controller for Mars capsule atmospheric entry using deep neural networks and flight-proven Apollo entry data. The controller is trained to modulate the bank angle with data from the Apollo entry simulations. The neural network controller reproduces the classical Apollo results over a variation of entry state initial conditions. Compared to the Apollo controller as a baseline, the present approach achieves the same level of accuracy for both linear and nonlinear entry dynamics. The Apollo-trained controller is then applied to Mars entry missions. As in Earth environment, the controller achieves the desired level of accuracy for Mars missions using both linear and nonlinear entry dynamics with higher uncertainties in the entry states and the atmospheric density. The deep neural network is only trained with data from Apollo reentry simulation in an Earth model and works in both Earth and Mars environments. It achieves the desired landing accuracy for a Mars capsule. This method works with both linear and nonlinear integration and can generate the bank angle commands in real-time without a prestored trajectory.

1. Introduction

Atmospheric entry requires a careful balance of landing accuracy, maximum acceleration, and heating requirements; however, landing accuracy is often sacrificed in order to meet other mission requirements [1]. While Earth reentry missions have become routine, Mars missions often do not have the landing accuracy desired. Additionally, Mars landings present unique challenges. Not counting the most recent InSight mission, there have been 20 missions that intended to land on Mars, of which only 7 were successful [2]. A summary of these missions is given in Table 1.

Most notably, the landing accuracy for the majority of these missions is on the order of hundreds of kilometers. To enable the next generation of Mars landers, future manned Mars missions will require a landing accuracy of tens of kilometers or even meters [3, 4]. In order to achieve that, several technical challenges need to be overcome. One of which is the capability to land accurately while maintaining acceptable acceleration and heating requirements set by the vehicle and cargo.

So far, there have been three generations of entry guidance algorithms. The first generation was designed for low-lifting capsule vehicles like Apollo. The Apollo entry guidance algorithm is broken up into different phases, e.g., preentry attitude hold, down control, up control, Kepler, and final entry [6, 7]. In each phase, the Apollo vehicle flies at a low angle of attack caused by the center of gravity offset with respect to its axis of symmetry, providing the lift for trajectory control [6]. This algorithm heavily depends on the given reference trajectory, and the command for the bank angle is generated to reduce the downrange error. To reduce the computational burden, this algorithm relies mostly on analytical, approximate, and empirical relationships, which affects the precision and applicability [8–13]. It is noteworthy that the Mars Science Laboratory (MSL) mission, which has the most accurate landing so far, used the Apollo reentry guidance algorithm to guide the vehicle to the parachute deployment velocity [14].

The second generation algorithm is the space shuttle entry guidance algorithm. The shuttle has an obviously higher lift-to-drag ratio than Apollo as well as longer flight

TABLE 1: Summary of previous successful Mars EDL missions [3, 5].

	Viking 1/2	MPF	MER-A/B	Phoenix	MSL
Year	1976	1997	2004/2008	2008	2010
Entry from	Orbit	Direct	Direct	Direct	Direct
Guidance	Unguided	Unguided	Unguided	Unguided	Apollo guidance
Entry velocity (m/s)	4700	7260	5400/5500	5500	5900
Entry flight path angle (deg)	-17	-14.06	-11.49/11.47	-13	-15.5
Ballistic coefficient (kg/m ²)	64	63	94	70	115
Entry mass (kg)	992	584	827/832	600	2920
Lift to drag ratio	0.18	0	0	0	0.24
Aeroshell diameter (m)	3.5	2.65	2.65	2.65	4.6
3- σ landing ellipse major Axis (km)	280	200	80	100	20
3- σ landing ellipse minor Axis (km)	100	100	12	21	10
Total integrated heating (J/m ²)	1100	3865	3687	2428	6185
Peak heating rate (W/cm ²)	26	100	44	58	155

time and downrange. Furthermore, the shuttle lands horizontally. In this acceleration-based entry guidance algorithm, a reference longitudinal trajectory is defined by a drag acceleration vs. Earth-relative velocity profile for high Mach numbers and a drag-vs.-energy profile for lower Mach numbers [8, 15, 16].

The third generation entry guidance algorithms in recent years originated from previous methods but rely much more on onboard computation for a real-time trajectory design and guidance solution. Predictor-corrector algorithms have shown great potential. These methods can adapt to large trajectory dispersions with no reliance on a given reference trajectory. Considering all their advantages and disadvantages, Lu developed a unified method based on the same algorithmic principles but applicable to multiple vehicle configurations [8]. The algorithm is relatively simple and highly robust using the bank angle to control the vertical component of the aerodynamic lift. Numerical predictor-corrector algorithms have also been adopted for various entry, descent, and landing scenarios, e.g., to address constraints [17] and aerocapture [18].

The Apollo reentry method is detailed in reentry guidance for Apollo by Morth [6]. It has been flight proven and is often the starting point of modern entry methods, including MSL [7, 19, 20]. On the other hand, even though several papers about MSL are published, the access to real Mars mission data is still limited. The large uncertainties in Mars atmosphere model should be considered which have significant influence on the results. Yu et al.'s review paper summarized the current navigation and guidance techniques for Mars pinpoint landing [2]. The paper indicated that the landing process is a critical and dangerous phase of the entire Mars landing mission leading to higher accuracy and safety requirements. The challenges for different phases including atmospheric entry, parachute descent, powered descent, and landing involve limited navigation information, nonlinearity and uncertainty of the dynamic model, and weak control capability in deep space.

Over the past few years, research using machine learning in spacecraft guidance and control was conducted. Recently, Biggs and Fournier used a recurrent neural network for state

propagation and cost function evaluation and then trained a multilayer perceptron (MLP) offline with optimal control data to achieve more fuel-efficient attitude control by thrusters [21]. Reinforcement learning for closed-loop control was applied to satellite rendezvous missions from current three-degree-of-freedom (3-DoF) dynamics to six-degrees-of-freedom (6-DoF) in the future [22]. Deep reinforcement learning (DRL) techniques were also used in onboard autonomous attitude guidance operations dealing with high-dimensional, continuous observation and action spaces [23]. In addition, machine learning algorithms were used in Space Situational Awareness sensor tasking of object observation and compared to the traditional methods [24].

For atmospheric entry, especially Mars entry, most of the methods are combinations of machine learning and other classical control techniques and set in idealized situations [25, 26]. For example, Jiang et al. combined reinforcement learning and Gauss pseudospectral methods in Mars powered descent entry [25], and Li and Jiang discussed second-order sliding mode guidance with radial basis function (RBF) neural network for Mars entry [26]. Gaudet et al. used deep reinforcement learning in a Mars powered descent and pinpoint landing problem, and it performed well with heuristic method for value function learning [27]. However, reinforcement learning has to care about both policy network and value network, which increases the complexity of the problem and the computation requirements when compared to deep neural networks.

Considering the constraints and the challenges discussed above, this work presents a simple and efficient method to control the vehicle during entry dealing with uncertainties in the entry state and the atmosphere. The proposed method has a strong advantage that it only needs the position to the target as an input to the controller to derive the bank angle command. A deep neural network is trained with Apollo simulation data, the most realistic data accessible, to learn the bank angle control commands. Then, the trained controller is used to replace the original control block of the Apollo simulation. After it has shown success in Apollo guidance [28], it is transferred to Mars environment with uncertainties

in the entry state and the atmosphere. Even though the guidance algorithm of Apollo was designed more than 50 years ago, it is still used as a reference for recent Mars missions [7, 19, 20], which motivates our approach for using it as the baseline for training the bank angle controller to guide the spacecraft in Mars atmosphere. Apollo capsule is a typical low-lifting vehicle with about 0.3 L/D ratio [6]. This neural network controller is designed for the same type of vehicle.

The paper is organized as follows. In Section 2, the general entry dynamics is introduced. Section 3 introduces the Apollo reentry case as the main reference of this work and then discusses the neural network setup and training process. After the discussion of the application of the trained controller and its performance in the Earth model in Section 3, Section 4 discusses the application and performance of the Mars model. The discussion of results, errors, and future improvements is presented in Section 5. Finally, concluding remarks are discussed in Section 6.

2. Entry Dynamics

The equations governing the entry of a capsule is given by [1, 29]

$$\begin{aligned}\dot{r} &= V \sin(\gamma), \\ \dot{\theta} &= \frac{V \cos(\gamma) \cos(\psi)}{r \cos(\phi)}, \\ \dot{\phi} &= \frac{V \cos(\gamma) \sin(\psi)}{r}, \\ \dot{V} &= -\frac{D}{m} - g \sin(\gamma), \\ \dot{\gamma} &= \frac{L}{mV} \cos(\sigma) - \frac{g}{V} \cos(\gamma) + \frac{V}{r} \cos(\gamma), \\ \dot{\psi} &= \frac{L \sin(\sigma)}{mV \cos(\gamma)} - \frac{V}{r} \cos(\gamma) \cos(\psi) \tan(\phi),\end{aligned}\quad (1)$$

where r is the distance from the center of the planet, V is the velocity of the capsule, γ is the flight path angle, ψ is the heading angle defined as the angle between the local parallel of latitude and the projection of velocity on the local horizontal plane [1, 29], ϕ is the latitude, θ is the longitude, m is the mass of the capsule, and σ is the roll/bank angle.

In this study, no constraints are imposed on σ . The drag D , lift L , and gravity g are given by

$$\begin{aligned}D &= 1/2 \rho V^2 C_D A, \\ L &= 1/2 \rho V^2 C_L A, \\ g &= \mu/r^2,\end{aligned}\quad (2)$$

where μ is the standard gravitational parameter; ρ is the atmospheric density; C_L and C_D are the coefficients of lift and drag, respectively; and A is the reference area used in calculating the aerodynamic forces. Both the nonspherical and the rotation effects are ignored as their effect on the dynamics

is considered marginal [30]. During entry, there is often only one control parameter, the bank angle; thus, the spacecraft flies in an “S” or a zigzag pattern so that it never deviates too far from its target laterally.

Classical guidance methods generally rely on a predetermined trajectory that the bank angle commands are generated to track. However, next generation EDL missions of low-lifting vehicles require that the control command be derived from onboard computation [31]. The objective of the present controller is to provide the bank angle commands to accurately land a Mars capsule based on flight-proven Apollo training sets. The trained neural network controller enables the guidance system to guide the vehicle without pre-stored reference trajectory, which is important for low-lifting vehicles with limited maneuverability. This algorithm is simple and efficient and shows robustness with respect to uncertainties in the initial states and atmospheric conditions in Mars atmospheric entry where the higher uncertainty presents a unique challenge for traditional controller design.

3. Apollo: Training and Verification

For Earth entry, the controller used for Apollo [6] is implemented to validate the model and provide training data for the neural network. The details of Apollo controller implementation in the Earth model were previously presented [28]. Simulations showed similar flight characteristics to the ones in [6]. Consistent with Apollo guidance [6], the following assumptions are made: point mass, constant mass, nonrotating planet, no thrust, drag acts in the direction opposite to velocity, lift is perpendicular to velocity, and gravity is directed along a vector from the point mass to the center of the planet. The Earth's atmosphere is represented with the standard atmospheric model used in Apollo [32]. The model is represented by three separate curves to represent the troposphere, lower stratosphere, and upper stratosphere. Following Apollo control, linear time integration is used to produce the training data while both linear and nonlinear integration methods are used in validation and application in order to test the robustness of the present controller.

3.1. Training the Deep Neural Network Based on Apollo. Artificial neural networks (ANN) are computing systems with interconnected groups of nodes inspired by the biological neural networks in animal brains [33]. Deep learning is the application of multineuron, multilayer neural networks which can perform regression, classification, clustering, and other tasks [34]. Since Hinton et al. published the famous deep belief network paper in 2006, deep learning technologies have shown their inherent capability of overcoming the drawback of traditional algorithms dependent on hand-designed features across disciplines [35, 36]. With machine learning techniques, computers have the capability of constructing algorithms that can learn from data and making data-driven decisions or predictions without being programmed explicitly [36]. Although recurrent neural network (RNN) is commonly used in time series related problems, our approach adopted feedforward neural network based method as it is simpler and more computationally tractable than

RNN for the full simulations of the dynamics. In this work, the neural networks are used to perform regression.

Neural networks work by using a number of nodes and node connections, where each of these connections has a weight. The simplest form of neural network is a feedforward neural network, where for a given set of inputs, there is an output without any loops or cycles. These neural networks are arranged in layers, where each layer contains a number of nodes. In neural networks, a sigmoid neuron, the most basic neuron, can be considered a single logistic node. Each one is connected to the input ahead of it, and a loss function is used to update the weights of the neuron as well as optimize the logistic fit to the incoming data [34]. In our training, we use the back propagation feedforward neural networks. The output of a given node, y , is obtained from

$$a = \sum_{i=1}^n w_i x_i + b$$

$$y = f(a) = f\left(\sum_{i=1}^n w_i x_i + b\right) \quad (3)$$

$$f(a) = \frac{2}{1 + e^{-2a}} - 1$$

where a is the net input, x_i represents the input to the node, w_i is the weight applied to the input, b is the bias term for the node, n is the number of inputs to the node, f is the hyperbolic tangent sigmoid function used as activation function to fit the ranges of input and output, [37, 38].

The backpropagation learning law to update the weights is also known as the “generalized delta rule,” which is the gradient descent method. In Equation (4), Δw_{ij} is used to update the weight w_{ij} between the output node i and input node j , $\eta > 0$ is the learning rate, δ_i is the error of the output node i and x_{ij} is the j th input to node i . The commonly used loss function is the mean squared error function [37–39].

$$\Delta w_{ij} = \eta \delta_i x_{ij}. \quad (4)$$

To start training the neural network, initial conditions for the nominal trajectory and physical characteristics obtained from the original Apollo report [6] are used as shown in Table 2. Aerodynamic coefficients including C_L and C_D are regarded as functions of Mach number as recorded in real Apollo flight [1]. To obtain the training data, the Apollo simulation was conducted for 216 runs with 500-meter variation in x , y , and z components of the initial position. The latitudes and longitudes of landing locations are shown in Figure 1(a), and the missed distances from the target are shown in Figure 1(b). The red filled point is the supposed target. From the plots, it is seen that all these points are close to the target and the missed distances are within the required range of Apollo mission guidance system, about 27 km [6].

Since our Apollo simulation with traditional controller using linear integration showed similar behavior to the real Apollo mission [28], the training dataset collected from the simulation with varying initial positions can be regarded as

TABLE 2: Initial conditions, physical characteristics, and target on Earth [1, 6].

Mass m (kg)	5498.22
Reference area A (m ²)	12.017
Height h (m)	121920
Velocity V (m/s)	11033.76
Flight path angle γ (deg)	-6
Range (m)	3704000
Latitude ϕ (deg)	-12.7
Longitude θ (deg)	122.9
Heading angle ψ (deg)	61
Terminated height (m)	7620
Target (m) in Earth-fixed frame	(-4754400.51, 3771311.48, 1964589.29)

the behavior of Apollo controller varying initial conditions. This dataset of 216 runs including 1669934 points from Apollo simulation is then used to train the neural network to improve the robustness of the new neural network controller.

In order to select the most critical parameters for training, mutual information is utilized. In probability and information theory, mutual information is a measure of the mutual dependence between two random variables. The maximal information coefficient (MIC) is a measure to apply mutual information on continuous random variables using binning [40]. Utilizing the software package minepy [41], MIC is calculated for the training dataset. The results show the strength of the linear or nonlinear association between variables including time, acceleration, position, velocity, distance between the current position and target, target location, and the bank angle. Five sets of 20000 randomly selected data points from the actual 216-run training trajectories are tested. The results are shown in Table 3.

Depending on the choice of dataset, the MIC values are different for the same combination of variables. However, time, distance, and velocity are more related to the bank angle since they generally have higher MIC and the MIC for acceleration and bank angle is always low. The target itself is out of the picture but distance considering target factor can be used instead of position since they have almost the same MIC values. As mentioned in Lu’s paper [31], time is usually not considered to be a critical parameter in entry flight. Using mutual information as a reference as well as training experience, we chose only distances between current positions and the target in x , y , and z directions as inputs to the neural network in training.

Based on the results obtained from MIC, three inputs of the training neural network represent 3 components in the distance from the current position to the target. The output neuron represents the one control variable, the bank angle. In training, the deep neural network is set with 10 layers including 8 hidden layers; the first layer includes 3 inputs; the 8 hidden layers contain 512, 256, 128, 64, 32, 16, 8, and 4 nodes, respectively; and the final output layer only has 1 output, the bank angle σ . Figure 2 depicts the structure used in training the neural network with 3 input nodes for the

FIGURE 1: Dataset generation with 500-meter variation in initial position.

TABLE 3: MIC to bank angle for selected training Apollo trajectories.

Variable	Time	Acceleration	Target	Pos-x	Pos-y	Pos-z	Dis-x	Dis-y	Dis-z	Vel-x	Vel-y	Vel-z
Set 1	0.8657	0.4452	0	0.8613	0.8614	0.8618	0.8613	0.86154	0.8618	0.8669	0.5484	0.8637
Set 2	0.8670	0.4492	0	0.8639	0.8639	0.8649	0.8639	0.8639	0.8649	0.8678	0.5496	0.8656
Set 3	0.8629	0.4468	0	0.8594	0.8591	0.8602	0.8594	0.8591	0.8602	0.8637	0.5480	0.8628
Set 4	0.8630	0.4434	0	0.8608	0.8608	0.8613	0.8608	0.8608	0.8613	0.8622	0.5423	0.8617
Set 5	0.8613	0.4385	0	0.8572	0.8572	0.8576	0.8572	0.8572	0.8576	0.8627	0.5453	0.8604

FIGURE 2: Neural network architecture example diagram.

distances from the target and 1 output node for the bank angle. This structure is started with 11 inputs and is constructed with a decreasing number of neurons on each layer

as commonly used in deep neural network. The structure of layers is kept the same in all trials in order to be flexible with varied number of inputs.

FIGURE 3: Applied neural network in control system.

TABLE 4: Linear integration using 10-layer NN in the Earth model.

Step	390	dt	2.0
Final position (m)	-4757796.00, 3771528.61, 1955895.88	Norm	6378603.26
Final velocity (m/s)	109.03, -63.50, -49.1	Norm	135.39
Target (m)	-4754400.51, 3771311.48, 1964589.29	Norm	6378614.71
Missed (m)	3395.49, 217.13, 8693.41	Norm	9335.51

(a) Flight characteristics

(b) Roll angle

FIGURE 4: Neural network controller in Earth model behavior.

The transfer functions connecting layers except the last layer are “hyperbolic tangent sigmoid.” The final/output layer is connected to the previous layer via “linear” transfer functions. Gradient descent is used as the training algorithm, and the full dataset feeds into training as a batch. The weights and biases are updated in the direction of the negative gradient of the selected performance function, which is the mean squared error function in this case. The goal is set to 0.01. The learning rate is set as 0.01, and 1000 epochs/iterations are conducted. Different results can be derived from different sets of training data with various numbers of runs and time steps. A 5-layer network and a 7-layer neural network includ-

ing less hidden layers and neurons were trained first. From numerical experiments, more neurons and more layers lead to generally better landing accuracy in application at the expense of longer training time as the number of neurons is a more important impact factor. Although increasing the number of layers and neurons can improve the training accuracy, it may also lead to overfitting which reduces the generalization capability in application. As a result, we finally adopt the 10-layer network with 512 neurons on the first layer.

The use of neural networks produces a controller that can learn to control a vehicle with a wide variety of vehicle

TABLE 5: Nonlinear integration using 10-layer NN in the Earth model.

Final position (m)	-4748517.48, 3786633.63, 1949302.58	Norm	6378620.00
Final velocity (m/s)	101.69, -83.58, -35.31	Norm	136.28
Target (m)	-4744398.27, 3791753.35, 1953372.11	Norm	6379840.99
Missed (m)	4119.21, 5119.72, 4069.53	Norm	7729.19

(a) Latitude and longitude distribution

(b) Missed distance

FIGURE 5: Monte Carlo simulation of nonlinear application in the Earth model varying initial positions.

characteristics in a variety of atmospheric conditions. For the simulations, it is assumed that the inputs and outputs in application are within the same range as inputs and outputs in training. As a result, the same normalization method is applied with the same maximum and minimum values. The neural network is applied as an alternative Apollo controller to test this new method and then adapted for Mars entry with more uncertainties.

3.2. Validation of Apollo Reentry. Using the same initial conditions of the Apollo case in Table 2, the neural network control is applied to the Earth entry problem. Linear integration is first implemented as in Apollo mission, and then, nonlinear integration is conducted. The trained deep neural network is used as a replacement of the original control block in this application. Figure 3 shows a simplified representation of how the neural network trained controller is called in the feedback loop to determine the bank angle input to the dynamics in Equation (1). The error is defined as the difference between the current state and the target state in Cartesian coordinates, $\mathbf{e}(t) = \mathbf{r}_T - \mathbf{r}(t)$, where $\mathbf{r}(t) = [x(t), y(t), z(t)]^T$ is the current position of the entry capsule. All other environmental parameters are kept the same except the control block which is removed.

First-order Taylor series approximation is used to propagate the dynamics as was done in [6]. The details of the results with 5-layer neural network and linear integration were discussed in [28]. The run of 10-layer neural network controller with the same target gives a final state as shown in Table 4. The 10-layer neural network leads to a landing as close as 9 km to the target in the Earth model. The simula-

TABLE 6: Statistics of norm of final missed distance for Monte Carlo simulation in the Earth model.

Varying	Initial position
Max (m)	56146
Min (m)	7639.9
Mean (m)	16418
Median (m)	14078
Standard deviation (m)	8581.5

TABLE 7: Confidence interval for Monte Carlo simulation in the Earth model.

Range	Number	Percentage
1-sigma	762	76.2%
2-sigma	952	95.2%
3-sigma	988	98.8%

tion trials all include the special case of lateral switches as designed in Apollo mission. The special case occurs in lateral logic when the roll angle is within 15 degrees; as a result, a constant 15-degree roll is called in response as in Apollo reentry [6].

The flight characteristics and roll angle behavior of the 10-layer neural network controller are shown in Figure 4. Figure 4(a) closely matches the Apollo case with the original control block in [6]; however, the curves are smooth as a result of the neural network training. During the application with linear integration, different time step sizes are tried to

FIGURE 6: Error ellipsoid projection of Monte Carlo simulation results in the Earth model.

find the best performance. The step size of 2 seconds leads to the best result since the computer logic of Apollo reentry is designed to update every 2 seconds [6].

Although the Apollo mission used linear integration, to achieve higher control accuracy and more potential applications, high-order numerical integration is adopted for the nonlinear entry dynamics in Equation (1). In controller applications with linear integration, the time step size is adjusted to meet the requirements following Apollo program. However, skipping the dynamic details may lead to inaccurate results. To get more accurate performance and more realistic simulation, the nonlinear dynamic model with higher order numerical integration (MATLAB ode45) is adopted.

The 10-layer neural network trained with 216-run dataset is used here since this one has the best performance with

TABLE 8: Initial conditions, physical characteristics [26], and target on Mars.

Mass m (kg)	2920
Reference area A (m ²)	16.619
Height h (m)	135400
Velocity V (m/s)	6000
Flight path angle γ (deg)	-11.5
Latitude ϕ (deg)	0
Longitude θ (deg)	0
Heading angle ψ (deg)	90
Terminated velocity	Mach 3
Target (m) in Mars-fixed frame	(3253225.27, 47680.98, 1023023.02)

TABLE 9: Linear integration using 10-layer NN in the Mars model.

Step	2383	dt	0.1
Final position (m)	3237005.19, 58912.16, 1020492.64	Norm	3394566.02
Final velocity (m/s)	-288.06, 398.50, 529.96	Norm	722.94
Target (m)	3253225.27, 47680.98, 1023023.02	Norm	3410619.33
Missed (m)	16220.08, 11231.18, 2530.38	Norm	19890.53

FIGURE 7: Neural network controller in Mars model behavior.

TABLE 10: Nonlinear integration with 10-layer NN in the Mars model.

Final position (m)	3236780.86, 58990.06, 1021338.16	Norm	3394607.75
Final velocity (m/s)	-286.78, 397.28, 531.65	Norm	723.00
Target (m)	3253225.27, 47680.98, 1023023.02	Norm	3410619.33
Missed (m)	16444.41, 11309.08, 1684.86	Norm	20028.79

linear integration. Table 5 shows that the neural network trained controller successfully drives the spacecraft with nonlinear integration in the Earth model with a missed distance of only 7.7 km from the target.

Although only distances between current positions and the target are used as neural network inputs, velocity is another critical parameter in entry flight control. From the results, both linear and nonlinear applications have a final velocity of about 136 m/s, close to the final velocity of the entry phase in Apollo mission. This further validates the neural network method.

Monte Carlo simulations are conducted to validate the method for the nonlinear dynamics with a 10-layer neural network in the Earth model. 1000 runs are generated with 3σ of 200 meters in x , y , and z directions. The results are shown in Figure 5. The statistics of performance and

confidence interval are summarized in Tables 6 and 7, respectively.

The corresponding projections of the error ellipsoid are plotted in Figure 6 with the modified Johnson function [42].

The results show that the range error of the final landing point with nonlinear integration is within 27 km range which is the design requirement for the Apollo capsule. The statistical error analysis shows that approximately 99% of the points are within the 3σ range. Additionally, the above results provided confidence in the neural network trained controller which is then applied to Mars entry dynamics.

4. Mars Entry Neural Network-Based Control

The uncertainties in the Mars atmosphere model are much larger than those in the Earth model. It is hard to find one

(a) Latitude and longitude distribution

(b) Missed distance

FIGURE 8: Monte Carlo simulation of nonlinear application in the Mars model varying initial positions.

(a) Latitude and longitude distribution

(b) Missed distance

FIGURE 9: Monte Carlo simulation of nonlinear application in the Mars model varying density.

accurate enough although there are several Mars atmospheric models available. Considering reasonable entry conditions like speed of sound and realistic Mars environment, an exponential atmospheric density model is adopted, Equation (5), and MSL mission environment is considered in this application [26, 43]. The density is calculated based on the density on the surface of Mars, the radius of Mars, the constant scale height, and current position.

$$\rho = \rho_0 e^{(-r-r_0)/h_s}, \quad (5)$$

where ρ is the density, $\rho_0 = 0.0158 \text{ kg/m}^3$ is the density on the surface of Mars, r is the current position, $r_0 = 3386.6 \text{ km}$ is

the radius of Mars, and $h_s = 9354.5 \text{ m}$ is the constant scale height.

Because of the complexity and uncertainties of current Mars atmosphere model, in this application, a constant temperature of 170 K is assumed during the entry process. In the trials, the effect due to the variation of temperature can be neglected since it is too small. The MSL data system flight result analysis showed that the temperature varied between about 150 K and 200 K which made this assumption reasonable [43].

The speed of sound is calculated based on ideal gas assumption. The MSL mission parameters and the simulation based on MSL are used as references when the trained neural network is applied to the Mars model since MSL capsule is also a low-lifting vehicle with a L/D ratio close to

Apollo's [26, 43, 44]. Table 8 shows the initial conditions and physical characteristics of the Mars capsule from the simulation in Li and Jiang and MSL report [26, 43]. Aerodynamic coefficients are set as having the same functions as in Apollo simulation. Velocity is used as the trigger to switch to descent phase from the entry phase at reasonable height as it is related to dynamic pressure. The value comes from the research of future Mars human mission considering the parachute design and other factors [45].

The same 10-layer neural network controller is applied in the Mars model with linear integration first. The reachable target is chosen based on the input and output ranges of the training set. The final state is shown in Table 9.

The flight characteristics and roll angle behavior of the 10-layer neural network controller with linear integration are shown in Figure 7.

It is seen that the capsule can land within 27 km designed as the requirement of track and range in Apollo guidance system with trained neural network controller [6]. The 10-layer neural network controller has successful landing in the Mars model with an error of 20 km. This is the best performed neural network with linear integration so far. Unlike in the Earth model with Apollo reference, the best time step size of the computer logic in the Mars model is worth investigating.

The 10-layer neural network controller is then applied to Mars nonlinear entry dynamics (Equation (1)). The performance of the 10-layer neural network controller in the Mars model is shown in Table 10.

The final landing location is still close to the target considering the scale. The missed distances are in a reasonable range which is about 20 km. The final positions with the same target are almost same in linear and nonlinear integration.

Monte Carlo simulations are conducted to validate the nonlinear method with 10-layer neural network in the Mars model. 1000 runs are generated with a 3σ of 500 meters in the entry state x , y , and z directions, and another 1000 runs are for varying density with a 3σ of 10%. The results varying initial positions are shown in Figure 8, and the results varying density are plotted in Figure 9. The statistics of the neural network performance and confidence interval are summarized in Tables 11 and 12, respectively. The corresponding projections of the error ellipsoids are shown in Figures 10 and 11, respectively, utilizing the modified Johnson's function [42].

The red dot shows the actual landing location with nominal trajectory. From the plots, the missed distances mostly fall around 27 km which is acceptable in Mars environment and the locations of final points are in good distribution.

From the statistics, both Monte Carlo simulations have good performance. The standard deviations are relatively small and the means as well as the medians are close to the missed distance of the nominal trajectory. The results show that about 98% of the points fall into the 3σ range of error.

Compared to the original uniform distribution of the Apollo dataset with the same variation in initial position, Figures 1(a) and 1(b), the Monte Carlo results are in similar distributions. Although all environmental parameters are changed to Mars atmosphere and the uncertainty has increased, these results show that the neural network control-

TABLE 11: Statistics of norm of final missed distance for Monte Carlo simulation in the Mars model.

Varying	Initial position	Density
Max (m)	27308	40632
Min (m)	19713	19616
Mean (m)	20467	21182
Median (m)	20122	20424
Standard deviation	994.0062	2109.2

TABLE 12: Confidence interval for Monte Carlo simulation in the Mars model.

Varying Range	Initial position		Density	
	Number	Percentage	Number	Percentage
1-sigma	884	88.4%	876	87.6%
2-sigma	952	95.2%	945	94.5%
3-sigma	976	97.6%	979	97.9%

ler can still work on Mars and land the capsule on target within the desired range. Considering the controller is trained by Earth reentry data only, this flexibility is notable as it provides a new approach for the control block design in new missions.

5. Discussion

The results presented show that the new method works in both Earth and Mars models with missed distances within the design requirement of the Apollo capsule. This result shows a more accurate landing than most of previous Mars missions. This method is also much simpler since the neural network can be trained on Earth before the mission and only needs positions with respect to the target as inputs. Considering the deep neural network is trained with data from Apollo simulation using the algorithm used in real Apollo missions, the simulations are more practical than in idealized situations. Additionally, the presented results show the applicability of the trained controller to the full nonlinear dynamics achieving accurate results for both Earth and Mars applications.

In order to improve the training performance, the training set is different from the reference and the application case in certain aspects. To get more data points and to help the neural network learn "how to react" better in all possible situations, the time step size used in the training set is as small as 0.1 seconds. In addition, the special case of lateral switches in the final phase is removed from the training set since the roll angle changing dramatically at switch points is hard to be captured by neural network learning, and added back later in Earth application. For Mars, the special case was not used and the controller was applied directly as was computed from the neural network.

Several sources of error affect the results presented. The most important one is lack of real data. Although the Apollo results are based on the real Apollo reentry algorithm which makes this research more practical, not all required information is available. Sensors and average-g method are not

FIGURE 10: Error ellipsoid projection of Monte Carlo simulation results in the Mars model varying initial positions.

applicable in this simulation. Additionally, the maximum and minimum values of inputs and outputs used in normalization are from Earth training data because there is no real Mars data available in this case. The reachable targets on Mars should be chosen based on Mars data.

Unlike the first generation controllers relying on pre-stored trajectories, the third generation controllers have an advantage in real-time implementation. Due to the nature of neural networks, the stability is hard to prove. However, the neural network controller is trained by Apollo flight data so it follows the stability rules of Apollo controller implicitly which accommodates biases in input signals to the steering

equations and allows the vehicle to alternate between 0 and 180-degree roll angle [6].

More work is needed to determine the feasibility of neural networks for future missions. A map of initial position covering the whole Earth is developed to obtain new training datasets. Grid training is attempted based on the mentioned map so that any point can be used as the initial position with a corresponding trained neural network. More accurate atmospheric models, Mars GRAM [46], can be used to have more representative results for the Mars environment. The terminal state including final velocity and height can be optimized based on specific mission requirements and design

FIGURE 11: Error ellipsoid projection of Monte Carlo simulation results in the Mars model varying density.

limits integrated with other subsystems. Aside from the addition of more complexity, transfer learning [47] is strongly considered since it may lead to better results in the Mars model with less time and investment. We conjecture that additional data from Mars missions, albeit limited, can significantly improve the landing accuracy and the overall flight performance of the spacecraft. Transfer learning can also be applied to missions with different types of vehicles. The corresponding neural network will be trained from the base neural network of low-lifting vehicle with a limited amount of the new vehicle flight data.

6. Conclusion

Although Mars entry problem is not a new area of research, the method of applying deep neural network as the main controller instead of approximating parameters is, to the best of our knowledge, novel. In this research, a case of 216 runs including 1669934 points from Apollo simulation is selected to train the neural network. The mutual information is calculated to decide input variables, which are the distances from the current position to the target in Cartesian coordinates. The neural network with 10 layers and 512 neurons on the

first layer reaches the best performance in application. It is applied as a replacement of the original control block and lands the capsule on Earth within a range of about 9 km with linearized dynamics and 8 km for the full nonlinear 3-DoF dynamics. As Apollo is a low-lifting vehicle, the trained neural network is supposed to be applied to a similar vehicle. The neural network is then adopted to the Mars model and lands the low-lifting capsule about 20 km around the target for both linearized and nonlinear dynamics. The statistical error analysis shows the robustness of this method to uncertainties in initial states as well as in atmospheric conditions with about 99% and 98% of points within acceptable range errors for both Earth and Mars applications, respectively. The results show that the deep neural network trained with Apollo data can replace the traditional control block to land the capsule on target in atmospheric entry. The successful performance of high-order numerical integration shows the potential of the present algorithm to handle modeling uncertainties as well as parameters and state uncertainties as shown in density and initial state variation results.

The deep neural network can work as a powerful controller to land the capsule on target during atmospheric entry. The high flexibility of the neural network application can be seen from the fact that the deep neural network is only trained by Earth reentry data and can work in both Earth and Mars environments with a similar vehicle. Furthermore, only the position with respect to the target is required as an input to the neural network so that it can be trained and implemented efficiently. This new method enables all bank angle search to be completed online without any reference trajectory even in large uncertainty environment. The present method is shown to be beneficial for future mission development with limited time and budget and can serve as a good baseline for future entry algorithms. The method shows a strong possibility that deep neural network can be used in future missions with large uncertainty as it is simple, efficient, and flexible and meets online computation requirements.

Data Availability

Data is available on request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

Article processing charges were provided in part by the UCF College of Graduates Studies Open Access Publishing Fund.

References

- [1] K. D. Hicks, "Introduction to astrodynamics reentry," Air Force Inst of Tech Wright-Patterson AFB OH Graduate School Of Engineering And Management, 2009, Tech. Rep. AFI-T/EN/TR-09-03.
- [2] Z. Yu, P. Cui, and J. L. Crassidis, "Design and optimization of navigation and guidance techniques for Mars pinpoint landing: Review and prospect," *Progress in Aerospace Sciences*, vol. 94, pp. 82–94, 2017.
- [3] R. D. Braun and R. M. Manning, "Mars exploration entry, descent, and landing challenges," *Journal of Spacecraft and Rockets*, vol. 44, no. 2, pp. 310–323, 2007.
- [4] M. B. Quadrelli, L. J. Wood, J. E. Riedel et al., "Guidance, navigation, and control technology assessment for future planetary science missions," *Journal of Guidance, Control, and Dynamics*, vol. 38, no. 7, pp. 1165–1186, 2015.
- [5] S. Li and X. Jiang, "Review and prospect of guidance and control for Mars atmospheric entry," *Progress in Aerospace Sciences*, vol. 69, pp. 40–57, 2014.
- [6] R. H. Morth, *Reentry Guidance for Apollo*, Massachusetts Institute of Technology, Apollo Guidance and Navigation, vol. 1, Massachusetts Institute of Technology Instrumentation Laboratory, Cambridge, MA, 1966.
- [7] G. Carman, D. Ives, and D. Geller, "Apollo-derived Mars precision lander guidance," in *23rd Atmospheric Flight Mechanics Conference*, p. 4570, Boston, MA, August 1998.
- [8] P. Lu, "Entry guidance: a unified method," *Journal of Guidance, Control, and Dynamics*, vol. 37, no. 3, pp. 713–728, 2014.
- [9] Z. R. Putnam, S. H. Bairstow, R. D. Braun, and G. H. Barton, "Improving lunar return entry range capability using enhanced skip trajectory guidance," *Journal of Spacecraft and Rockets*, vol. 45, no. 2, pp. 309–315, 2008.
- [10] C. W. Brunner and P. Lu, "Skip entry trajectory planning and guidance," *Journal of Guidance, Control, and Dynamics*, vol. 31, no. 5, pp. 1210–1219, 2008.
- [11] C. W. Brunner and P. Lu, "Comparison of fully numerical predictor-corrector and Apollo skip entry guidance algorithms," *The Journal of the Astronautical Sciences*, vol. 59, no. 3, pp. 517–540, 2012.
- [12] M. Tigges, T. Crull, J. Rea, and W. Johnson, "Numerical skip-entry guidance," in *29th Annual AAS Guidance and Control Conference*, Breckenridge, Colorado, 2006.
- [13] S. H. Bairstow, *Reentry guidance with extended range capability for low L/D spacecraft*, [Master's thesis], MIT, Cambridge, MA, 2006.
- [14] M. S. Martin, G. F. Mendeck, P. B. Brugarolas et al., "In-flight experience of the Mars science laboratory guidance, navigation, and control system for entry, descent, and landing," *CEAS Space Journal*, vol. 7, no. 2, pp. 119–142, 2015.
- [15] J. C. Harpold and C. A. Graves Jr., "Shuttle entry guidance," in *25th American Astronautical Society Anniversary Conference*, Houston, TX, 1978.
- [16] J. C. Harpold and D. E. Gavert, "Space shuttle entry guidance performance results," *Journal of Guidance, Control, and Dynamics*, vol. 6, no. 6, pp. 442–447, 1983.
- [17] Y. Zheng, H. Cui, and Y. Ai, "Constrained numerical predictor-corrector guidance for Mars precision landing," *Journal of Guidance, Control, and Dynamics*, vol. 40, no. 1, pp. 179–187, 2017.
- [18] P. Lu, C. J. Cerimele, M. A. Tigges, and D. A. Matz, "Optimal aerocapture guidance," *Journal of Guidance, Control, and Dynamics*, vol. 38, no. 4, pp. 553–565, 2015.
- [19] G. Mendeck and L. Craig, "Entry guidance for the 2011 Mars Science Laboratory mission," in *AIAA Atmospheric Flight Mechanics Conference*, Portland, Oregon, August 2011.
- [20] A. M. San Martin, S. W. Lee, and E. C. Wong, "The development of the MSL guidance, navigation, and control system

- for entry, descent, and landing,” in *23rd Space Flight Mechanics Meeting*, Kauai, Hawaii, 2013.
- [21] J. D. Biggs and H. Fournier, “Neural-network-based optimal attitude control using four impulsive thrusters,” *Journal of Guidance, Control, and Dynamics*, vol. 43, no. 2, pp. 299–309, 2020.
 - [22] J. Broida and R. Linares, “Spacecraft rendezvous guidance in cluttered environments via reinforcement learning,” *Advances in Astronautical Sciences: AAS/AIAA Space Flight Mechanics Meeting*, vol. 168, pp. 1776–1788, 2019.
 - [23] A. Harris and H. Schaub, “Deep on-board scheduling for autonomous attitude guidance operations,” in *AAS Guidance, Navigation and Control Conference*, Breckenridge, CO, 2020AAS 02-117.
 - [24] B. D. Little and C. E. Frueh, “Space situational awareness sensor tasking: comparison of machine learning with classical optimization methods,” *Journal of Guidance, Control, and Dynamics*, vol. 43, no. 2, pp. 262–273, 2020.
 - [25] X. Jiang, S. Li, and R. Furfaro, “Integrated guidance for Mars entry and powered descent using reinforcement learning and pseudospectral method,” *Acta Astronautica*, vol. 163, pp. 114–129, 2019.
 - [26] S. Li and X. Jiang, “RBF neural network based second-order sliding mode guidance for Mars entry under uncertainties,” *Aerospace Science and Technology*, vol. 43, pp. 226–235, 2015.
 - [27] B. Gaudet, R. Linares, and R. Furfaro, “Deep reinforcement learning for six degree-of-freedom planetary landing,” *Advances in Space Research*, vol. 65, no. 7, pp. 1723–1741, 2020.
 - [28] H. Wang, D. Martin, and T. A. Elgohary, “Neural network trained controller for atmospheric entry in Mars missions,” *Advances in Astronautical Sciences: AAS/AIAA Space Flight Mechanics Meeting*, vol. 168, pp. 2477–2496, 2019.
 - [29] N. X. Vinh, A. Busemann, and R. D. Culp, *Hypersonic and Planetary Entry Flight Mechanics*, University of Michigan Press, Ann Arbor, MI, 1980.
 - [30] G. L. Jacob, G. Neeler, and R. V. Ramanan, “Mars entry mission bank profile optimization,” *Journal of Guidance, Control, and Dynamics*, vol. 37, no. 4, pp. 1305–1316, 2014.
 - [31] P. Lu, “Predictor-corrector entry guidance for low-lifting vehicles,” *Journal of Guidance, Control, and Dynamics*, vol. 31, no. 4, pp. 1067–1075, 2008.
 - [32] N. Sissenwine, M. Dubin, and H. Wexler, “The U.S. standard atmosphere, 1962,” *Journal of Geophysical Research*, vol. 67, no. 9, pp. 3627–3630, 1962.
 - [33] M. van Gerven and S. Bohte, “Editorial: Artificial neural networks as models of neural information processing,” *Frontiers in Computational Neuroscience*, vol. 11, no. 114, 2017.
 - [34] W. G. Hatcher and W. Yu, “A survey of deep learning: platforms, applications and emerging research trends,” *IEEE Access*, vol. 6, pp. 24411–24432, 2018.
 - [35] G. E. Hinton, S. Osindero, and Y.-W. Teh, “A fast learning algorithm for deep belief nets,” *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, 2006.
 - [36] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, and F. E. Alsaadi, “A survey of deep neural network architectures and their applications,” *Neurocomputing*, vol. 234, pp. 11–26, 2017.
 - [37] B. J. Wythoff, “Backpropagation neural networks: A tutorial,” *Chemometrics and Intelligent Laboratory Systems*, vol. 18, no. 2, pp. 115–155, 1993.
 - [38] J. Kamruzzaman and S. Aziz, “A note on activation function in multilayer feedforward learning,” in *Proceedings of the 2002 International Joint Conference on Neural Networks. IJCNN’02 (Cat. No. 02CH37290)*, pp. 519–523, Honolulu, HI, USA, USA, May 2002.
 - [39] P. Kim, *Matlab Deep Learning, With Machine Learning, Neural Networks and Artificial Intelligence*, Springer, New York, NY, 2017.
 - [40] J. B. Kinney and G. S. Atwal, “Equitability, mutual information, and the maximal information coefficient,” *Proceedings of the National Academy of Sciences*, vol. 111, no. 9, pp. 3354–3359, 2014.
 - [41] D. Albanese, M. Filosi, R. Visintainer, S. Riccadonna, G. Jurman, and C. Furlanello, “Minerva and minepy: a C engine for the MINE suite and its R, Python and MATLAB wrappers,” *Bioinformatics*, vol. 29, no. 3, pp. 407–408, 2013.
 - [42] A. Johnson, “error_ellipse,” *MathWorks File Exchange*, 2004, Nov. 2019, https://www.mathworks.com/matlabcentral/fileexchange/4705-error_ellipse.
 - [43] C. Karlgaard, P. Kutty, J. Shidner, M. Schoenenberger, and M. Munk, “Mars entry atmospheric data system trajectory reconstruction algorithms and flight results,” in *51st AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*, p. 28, Grapevine (Dallas/Ft. Worth Region), TX, January 2013.
 - [44] J. M. Lafleur and C. J. Cerimele, “Mars entry bank profile design for terminal state optimization,” *Journal of Spacecraft and Rockets*, vol. 48, no. 6, pp. 1012–1024, 2011.
 - [45] J. A. Christian, G. Wells, J. M. Lafleur, A. Verges, and R. D. Braun, “Extension of traditional entry, descent, and landing technologies for human Mars exploration,” *Journal of Spacecraft and Rockets*, vol. 45, no. 1, pp. 130–141, 2008.
 - [46] C. G. Justus, B. F. James, S. W. Bougher et al., “Mars-GRAM 2000: a Mars atmospheric model for engineering applications,” *Advances in Space Research*, vol. 29, no. 2, pp. 193–202, 2002.
 - [47] S. J. Pan and Q. Yang, “A survey on transfer learning,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2010.