Six-degree-of-freedom Optimal Feedback Control of Pinpoint Landing using Deep Neural Networks

Omkar S. Mulekar

Hancheol Cho

Riccardo Bevilacqua

Six-degree-of-freedom Optimal Feedback Control of Pinpoint Landing using Deep Neural Networks

Omkar S. Mulekar *
University of Florida, Gainesville, FL

Hancheol Cho† and Riccardo Bevilacqua‡
Embry-Riddle Aeronautical University, Daytona Beach, FL

Machine learning regression techniques have shown success at feedback control to perform near-optimal pinpoint landings for low fidelity formulations (e.g. 3 degree-of-freedom). Trajectories from these low-fidelity landing formulations have been used in imitation learning techniques to train deep neural network policies to replicate these optimal landings in closed loop. This study details the development of a near-optimal, neural network feedback controller for a 6 degree-of-freedom pinpoint landing system. To model disturbances, the problem is cast as either a multi-phase optimal control problem or a triple single-phase optimal control problem to generate examples of optimal control through the presence of disturbances. By including these disturbed examples and leveraging imitation learning techniques, the loss of optimality is reduced for pinpoint landing scenario.

I. Nomenclature

$\mathbf{x}$	=	Lander state
$\mathbf{u}$	=	Control input
$\mathbf{u}^*$	=	Optimal control input
l	=	Policy loss function
θ	=	Policy parameters or pitch angle
π	=	Policy function
J	=	Optimal control objective function
$\mathbf{f}$	=	Function defining equations of motion
$\mathbf{g}$	=	Function defining path inequality and equality constraints
$\mathbf{e}$	=	Function defining endpoint constraints
L	=	Optimal control path cost function
Φ	=	Optimal control terminal cost function
M	=	Number of phases
ϕ_i	=	Multi-phase optimal control problem transition functions

II. Introduction

Space agencies around the globe are making active efforts to deliver payloads to the surface of the Moon, Mars, and other terrestrial bodies to collect scientific data, perform studies, and expand human presence [1–3]. Robotic surface missions to the moon typically consist of a soft lander and a robot with data-collecting instruments.

There have been efforts to develop optimal feedback controllers for the pinpoint landing problem. As computing power and use of Graphic Processing Units (GPUs) have increased, they have been leveraged to apply machine learning techniques to train highly parametrized regression techniques like the deep neural network (DNN) to learn feedback

*PhD Candidate, Department of Mechanical and Aerospace Engineering, University of Florida, Gainesville, FL 32611

†Assistant Professor, Department of Aerospace Engineering, 1 Aerospace Blvd, Embry-Riddle Aeronautical University, Daytona Beach, FL 32114

‡Professor, Department of Aerospace Engineering, Embry-Riddle Aeronautical University, 1 Aerospace Blvd, Daytona Beach, FL 32114. AIAA Associate Fellow

control in many applications such as image recognition, speech recognition, and robotic control [4–6]. Optimal control problems are generally formulated to minimize an integral objective function subject to various differential constraints (ODEs), algebraic equality constraints, and algebraic inequality constraints[7].

Several studies have made use of these numerical methods to generate open-loop optimal trajectories for low-fidelity formulations of the landing problem, and then imitation learning techniques have been used to train policies to duplicate these trajectories in closed loop. These policies are often represented by deep neural networks, and are trained via behavioral cloning or Dataset Aggregation (DAgger) [8–11]. Deep neural networks are one type of parametric regression technique, i.e. they perform regression by tuning parameters (weights and biases) that define the model [12]. In the realm of optimal landings, there has been limited work on maintaining optimality through a disturbance in the trajectory.

The objective of this research is to train a neural network to perform optimal feedback control of a 6 Degree-of-Freedom (6DOF) pinpoint lander. By leveraging numerical optimization techniques and modeling disturbances as either multi-phase optimal control or triple single-phase optimal control problems, a large dataset of open-loop optimal trajectories is generated. Behavioral cloning is then employed to train the neural network to perform robust feedback control and to mitigate loss of optimality in the presence of a constant, time-bound disturbance.

This paper first gives an overview of imitation learning, optimal controls, and their applications to the pinpoint landing problem in Section III. Then, the formulations of interest and details on the policy and training techniques are provided in Section IV. The performances of the trained policies are given and discussed in Section V. Finally, the paper is concluded in Section VI.

III. Background

A. Imitation Learning

Imitation learning is a machine learning technique that involves training a policy to replicate the behavior of an "expert" by leveraging provided demonstratons [13]. In many applications, agents are trained to mimic human behavior, often in a feedback loop. Imitation learning strategies have been investigated for self-driving vehicles, tasks for robotic arms, and video games [11, 14, 15]. Although imitation learning techniques have also been used to mimic the behavior of non-human experts, e.g. when optimal control is the desired behavior, the core ingredients that make up an imitation learning method are the same, and considerable success has been seen in the use of these techniques in the aerospace domain.

The open-loop optimal trajectories are typically provided as a set of N ordered state-action pairs $\{(\mathbf{x}_1, \mathbf{u}_1^*), (\mathbf{x}_2, \mathbf{u}_2^*), ..., (\mathbf{x}_N, \mathbf{u}_N^*)\}$, as well as environment that defines a state given an action in the previous state. The policy class, π_θ for parametric or π for nonparametric, is a classification or regression technique that calculates an action $\mathbf{u}$ given an input state $\mathbf{x}$. The objective is to train a policy to minimize a loss function l that measures the deviation of policy predictions from expert behavior $\mathbf{u}^* = \pi^*(\mathbf{x})$. One common example of a loss function is the squared error $l(\mathbf{u}_1, \mathbf{u}_2) = (\mathbf{u}_1 - \mathbf{u}_2)^T(\mathbf{u}_1 - \mathbf{u}_2)$. For parametric policy classes, this objective is formulated as $\arg\min_\theta \mathrm{E}[l(\pi^*(\mathbf{x}), \pi_\theta(\mathbf{x}))]$ where E is the expected value given that states are sampled from *policy* behavior [16].

Behavioral cloning is the simplest imitation learning technique, which applies supervised learning on the provided example state-action pairs. Training via behavioral cloning does not make use of interaction with the expert, nor does it require rollout of the policy in the environment during training. For parametric policy classes, it is formulated as $\arg\min_\theta \mathrm{E}[l(\mathbf{u}^*, \pi_\theta(\mathbf{x}))]$ where E is the expected value given that states are sampled from *expert* behavior [16].

B. Optimal Control

This study makes use of both single-phase and multi-phase optimal control problems. The single-phase formulation for the optimal control problem can be written as

$$\begin{aligned} \min_{\mathbf{u}(t), t_f} \quad & J(\mathbf{x}(t), \mathbf{u}(t), t_f) \\ \text{subject to} \quad & \\ \dot{\mathbf{x}}(t) &= \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), t) \\ \mathbf{g}(\mathbf{x}(t), \mathbf{u}(t), t) &\leq \mathbf{0} \\ \mathbf{e}(\mathbf{x}(t_0), t_0, \mathbf{x}(t_f), t_f) &= \mathbf{0} \end{aligned} \tag{1}$$

Here, J is a cost functional defined as

$$J(\mathbf{x}(t), \mathbf{u}(t), t_f) = \int_{t_0}^{t_f} L(\mathbf{x}(t), \mathbf{u}(t), t)dt + \Phi(\mathbf{x}(t_f), \mathbf{u}(t_f), t_f) \tag{2}$$

where L is a path cost, Φ is a terminal cost, $\mathbf{f}$ defines the system equations of motion, $\mathbf{g}$ defines algebraic equality and inequality constraints, and $\mathbf{e}$ defines endpoint constraints [7]. Here, the initial time t_0 is fixed, and the final time t_f is free to be optimized.

Direct numerical methods are used to find the optimal control $\mathbf{u}^*(t)$ and the optimal trajectory of states $\mathbf{x}^*(t)$. We use the MATLAB package OpenOCL, which transcribes optimal control problems to nonlinear programming problems (NLP) and numerically solves them using IPOPT [17–19].

If equations of motion, cost functions, or constraints change at different points in the trajectory (as is the case when disturbances are present), multi-phase optimal control problems can be formulated. These multi-phase optimal control problems are formulated as follows

$$\begin{aligned}
&\min_{\substack{\mathbf{u}_1(t),\ldots,\mathbf{u}_M(t);\\ t_{f,1},\ldots,t_{f,M}}} \sum_{i=1}^{M} J_i(\mathbf{x}_i(t), \mathbf{u}_i(t), t_{f,i}) \\
&\text{subject to} \\
&\quad \text{for } i \in \{1, ..., M\} \\
&\qquad \dot{\mathbf{x}}_i(t) = \mathbf{f}_i(\mathbf{x}_i(t), \mathbf{u}_i(t), t) \\
&\qquad \mathbf{g}_i(\mathbf{x}_i(t), \mathbf{u}_i(t), t) \leq \mathbf{0} \\
&\qquad \mathbf{e}_i(\mathbf{x}_i(t_{0,i}), t_{0,i}, \mathbf{x}_i(t_{f,i}), t_{f,i}) = \mathbf{0} \\
&\quad \text{for } i \in \{1, ..., M-1\} \\
&\qquad \phi_i(\mathbf{x}_{i+1}(t_{0,i+1}), \mathbf{x}_i(t_{f,i})) = \mathbf{0}
\end{aligned} \tag{3}$$

where M is the number of phases, $t_{0,i}$ and $t_{f,i}$ are the initial and final times for phase i, respectively, and ϕ_i is a function that defines the transition between phases i and $i+1$. Since, for different phases, different equations of motion can be defined, disturbances can be modeled into the multi-phase optimal control problem.

C. Applications to Pinpoint Landing

Several studies have made use of imitation learning to train policies to perform maneuvers in a variety of landing formulations and scenarios. Sanchéz et al. generated optimal trajectories using direct methods, providing tens of millions of state action pairs for a neural network to learn from for a low degree-of-freedom multicopter [20]. In a similar study on the same formulation, the same authors used indirect methods to generate optimal trajectories, reducing chatter in the optimal control [21]. Furfaro et al. combined these optimal demonstrations with a lunar surface image generator that provided a lunar surface image given a lander state in order to train the feedback controller to take surface images as inputs rather than the state directly [9]. In the reinforcement learning realm, Gaudet et al. approached the 6DOF fuel-optimal pinpoint landing problem by using the Advantage-Actor-Critic algorithm [22].

IV. Approach

A. Single Phase Formulation (SPOCP)

For the 6DOF lander problem in this study, $\mathbf{x} = [\mathbf{r}^T, \dot{\mathbf{r}}^T, \mathbf{\Phi}^T, \boldsymbol{\omega}^T, m]^T$ and $\mathbf{u} = [u_1, u_2, u_3, u_4]^T$, where $\mathbf{r} = [x, y, z]^T$ is the position vector, $\mathbf{\Phi} = [\phi, \theta, \psi]^T$ are the Euler angles corresponding to roll, pitch, and yaw, respectively, and $\boldsymbol{\omega} = [p, q, r]^T$ are the angular velocities corresponding to rotations about the x, y, and z axes, respectively, and m is the

mass of the lander.

$$\min_{\mathbf{u}(t),t_f} \int_{t_0}^{t_f} (u_1^2 + k(u_2^2 + u_3^2 + u_4^2))dt$$

subject to

$$\begin{aligned}
\ddot{\mathbf{r}} &= \frac{1}{m} u_1 \hat{\mathbf{t}}_\phi - \mathbf{g} \\
\dot{\phi} &= p + q \sin\phi \tan\theta + r \cos\phi \tan\theta \\
\dot{\theta} &= q \cos\phi - r \sin\phi \\
\dot{\psi} &= q \sin\phi \sec\theta + r \cos\phi \sec\theta \\
\dot{p} &= \frac{1}{I_x}[u_2 + (I_y - I_z)qr] \\
\dot{q} &= \frac{1}{I_y}[u_3 + (I_z - I_x)rp] \\
\dot{r} &= \frac{1}{I_z}[u_4 + (I_x - I_y)pq] \\
\dot{m} &= -\frac{\sqrt{u_1^2 + k(u_2^2 + u_3^2 + u_4^2)}}{I_{sp} g_0}
\end{aligned} \tag{4}$$

where

$$\hat{\mathbf{t}}_\phi = \begin{bmatrix} \sin\phi \sin\psi + \cos\phi \sin\theta \cos\psi \\ -\sin\phi \cos\psi + \cos\phi \sin\theta \sin\psi \\ \cos\phi \cos\theta \end{bmatrix}$$

represents the axial vector that points up the lander body, and the gravity vector $\mathbf{g} = [0, 0, g]^T$ is earth's gravity (i.e. $g = 9.81$ m/s^2). The end point constraints contain the initial conditions $\mathbf{x}(t_0) = \mathbf{x}_0$ and the final conditions $\mathbf{r}(t_f) = \mathbf{r}_f = [0, 0, 0.1]^T$ m, $\dot{\mathbf{r}}(t_f) = \dot{\mathbf{r}}_f = [0, 0, -0.1]^T$ m/s, $\mathbf{\Phi}(t_f) = \mathbf{\Phi}_f$, $\boldsymbol{\omega}(t_f) = \boldsymbol{\omega}_f$. The final mass m is unconstrained and free to be optimized, though it should remain positive. The control u_1 is an axial force through the lander body in the direction of $\hat{\mathbf{t}}_\phi$. The controls u_2, u_3, and u_4 are applied moments to the lander. Each can be considered to be caused by two opposing forces $u_i/2$ for $i \in \{2, 3, 4\}$ placed a distance of 1 m on both sides of the lander center of mass. The constant $k = 1$ N/(N*m) in the cost function serves as a unit conversion. A diagram of the control forces applied to the 6DOF lander can be seen in Fig. 1.

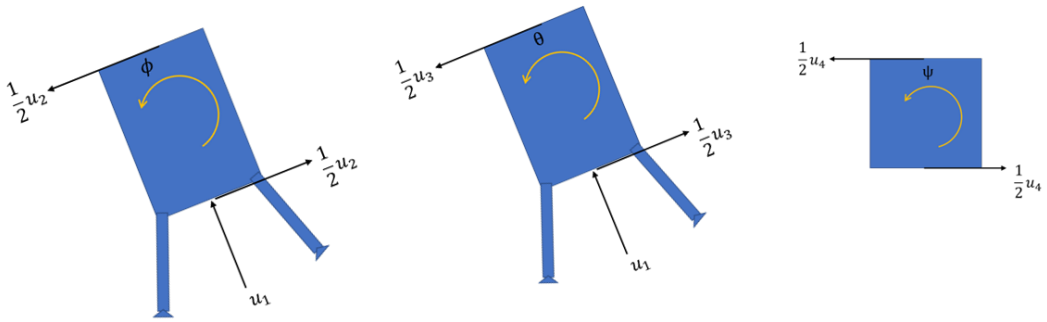

Fig. 1 Diagram of 6DOF lander

For this problem, there are no path constraints on the lander state, though common constraints on lander state like

glide-slope constraints can be included if needed for future studies. The constraints on the lander control are given as

$$\begin{aligned} 0 \leq & u_1 \leq 20 \text{ N} \\ -20 \text{ Nm} \leq & u_i \leq 20 \text{ Nm for } i \in \{2,3,4\} \end{aligned}$$

B. Multi-Phase Formulation (MPOCP)

The solution to the single-phase optimal control problem provides an optimal trajectory $\mathbf{x}^*$ and optimal control history $\mathbf{u}^*$ in the nominal case without a disturbance. However, an optimal trajectory for the off-nominal case of a disturbance applied at time $t \in [t_{0,d}, t_{f,d}]$ can be solved from a multi-phase optimal control problem (MPOCP) with three phases: *Phase 1* which uses nominal dynamics, *Phase 2* which uses the disturbed dynamics, and *Phase 3* which again uses the nominal dynamics. Each phase uses the same cost function $J_1 = J_2 = J_3 = J$.

Phases 1 and 3 use the same equations of motion, i.e. $\mathbf{f}_1 = \mathbf{f}_3 = \mathbf{f}$. However, since disturbances are applied during Phase 2, $\mathbf{f}_2 \neq \mathbf{f}_1$. Simply, the disturbance is applied as a constant added to the acceleration equations defining $\ddot{\mathbf{r}}$

$$\mathbf{f}_2(\mathbf{x}, \mathbf{u}, t) = \mathbf{f}_1(\mathbf{x}, \mathbf{u}, t) + \mathbf{d}$$

for $t \in [t_{0,2}, t_{f,2}]$, and where $\mathbf{d} = [\mathbf{0}_{1\times 3}, d_{\dot{x}}, d_{\dot{y}}, \mathbf{0}_{1\times 8}]^T$. The disturbances $d_{\dot{x}}$ and $d_{\dot{y}}$ are constants. The transition functions ϕ_i are defined as

$$\begin{aligned} &\textit{Phase 1-2: } \mathbf{x}_2(t_{0,2}) - \mathbf{x}_1(t_{f,1}) = \mathbf{0} \\ &\textit{Phase 2-3: } \mathbf{x}_3(t_{0,3}) - \mathbf{x}_2(t_{f,2}) = \mathbf{0} \end{aligned} \tag{5}$$

The same path constraints detailed in the single-phase formulation are kept for each phase of the multi-phase formulation. The initial condition for Phase 1 and the final condition for Phase 3 come from the same initialization as the initial condition and final condition for the single-phase formulation.

C. Triple Single-Phase Formulation (3SPOCP)

While the MPOCP fits well to problems where the dynamics change at specified points in the trajectory, when applied to the disturbed landing problem in the context of feedback control, it implies that the disturbance is known at initialization. A triple single-phase optimal control problem (3SPOCP) formulation is proposed here to demonstrate the case where the disturbance is not to be known at initialization. Then, a comparison of the SPOCP, MPOCP, and 3SPOCP is done.

Similar to the MPOCP, the 3SPOCP uses three equations of motion $\mathbf{f}_1$, $\mathbf{f}_2$, and $\mathbf{f}_3$ such that $\mathbf{f}_1 = \mathbf{f_3}$ and $\mathbf{f}_2 = \mathbf{f}_1 + \mathbf{d}$. The 3SPOCP trajectory generation is acheived in four steps, each using the same cost function J:

1) Generate an SPOCP trajectory using nominal dynamics with no disturbance $\dot{\mathbf{x}} = \mathbf{f}_1(\mathbf{x}, \mathbf{u}, t)$. Call the resulting state and control history $\mathbf{x}_1(t)$ and $\mathbf{u}_1(t)$.
2) Use the state in the first trajectory at time $t = t_{0,d}$ as the initial condition for an SPOCP using off-nominal dynamics with the disturbance $\dot{\mathbf{x}} = \mathbf{f}_2(\mathbf{x}, \mathbf{u}, t)$. Call the resulting state and control history $\mathbf{x}_2(t)$ and $\mathbf{u}_2(t)$.
3) Use the state in the second trajectory at time $t_{f,d}$ as the initial condition for an SPOCP using nominal dynamics with no disturbance $\dot{\mathbf{x}} = \mathbf{f}_3(\mathbf{x}, \mathbf{u}, t)$. Call the resulting state and control history $\mathbf{x}_3(t)$ and $\mathbf{u}_3(t)$.
4) The 3SPOCP trajectory can be arranged from the three SPOCP problems in a piecewise fashion as

$$\mathbf{x}(t) = \begin{cases} \mathbf{x}_1(t) & t < t_{0,d} \\ \mathbf{x}_2(t) & t_{0,d} \leq t \leq t_{f,d} \\ \mathbf{x}_3(t) & t_{f,d} > t \end{cases} \tag{6}$$

$$\mathbf{u}(t) = \begin{cases} \mathbf{u}_1(t) & t < t_{0,d} \\ \mathbf{u}_2(t) & t_{0,d} \leq t \leq t_{f,d} \\ \mathbf{u}_3(t) & t_{f,d} > t \end{cases} \tag{7}$$

D. Policy and Training

The policy class used to perform feedback control of the lander in this study is a DNN implemented using TensorFlow [23]. Three policies are trained in this study: one on solely SPOCP trajectories, one on a combination of

SPOCP and MPOCP trajectories, and one on a combination of SPOCP and 3SPOCP trajectories. We show that the nominal, single-phase optimal control of the 6DOF lander (as formulated in Eqn. 4) can be performed by a DNN with a fairly simple architecture trained on the SPOCP trajectories. The DNNs used in this study contain 5 fully connected hidden layers and an output layer. The hidden layers use a hyperbolic tangent activation function $a(z) = \tanh(z)$, and the output layer uses a linear activation function $a(z) = z$. The hidden layers each contain 600 neurons. Given the example state-action pairs generated in OpenOCL, behavioral cloning (i.e. supervised learning) is used to train the DNN. The training loss function is the mean squared error with a batch size of 100, and the Adam optimization method is used to train the DNN [24].

The first policy is trained on 20,000 SPOCP trajectories containing 2,000,000 state-action pairs. The second policy is trained on the same 20,000 SPOCP trajectories as well as 100 MPOCP trajectories containing 30,000 state-action pairs (300 pairs per trajectory), for a combined total of 2,030,000 state-action pairs. The third policy is trained on the same 20,000 SPOCP trajectories and 100 3SPOCP trajectories containing 30,000 state-action pairs (again, 300 pairs per trajectory), for a combined total of 2,030,000 state-action pairs.

V. Policy Performances

A. Nominal Performance

The policy trained solely on SPOCP trajectories was implemented in a closed-loop simulation. A block diagram of the policy implemented in a feedback loop can be seen in Fig. 2.

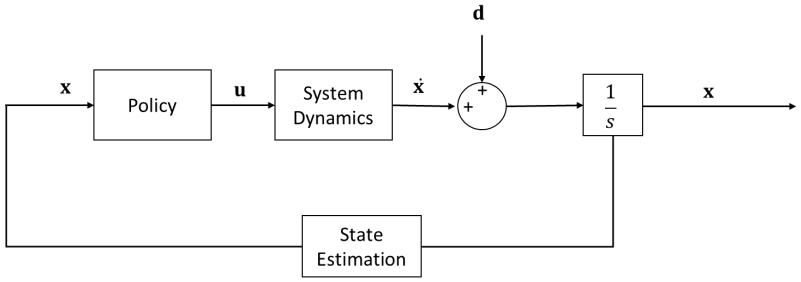

Fig. 2 Block diagram of implemented policy in closed loop with applied disturbances

A plot of the policy-driven trajectory in nominal dynamics (i.e. no applied disturbance) is shown in Fig. 3. The costs of the nominal trajectories are shown in Tab. 1.

From visual inspection, the policy-driven trajectory follows the open-loop optimal trajectory closely, though there is a 4.4% increase in the evaluated cost as seen in Tab. 1. While the ability to perform feedback near-optimal control of a 6DOF lander alone represents a significant achievement, there is interest in assessing (1) the stability and robustness of the controller, and (2) the ability of the controller to maintain optimality in the presence of disturbances (i.e. an off-nominal trajectory). This paper primarily focuses on the latter assessment.

To see the performance of the policy in an off-nominal trajectory, a disturbance was applied as an added acceleration. The trained policy was simulated for an off-nominal case with an added disturbance of $d_{\dot{x}} = -d_{\dot{y}} = 1$ m/s^2 for $0.9 \text{ s} \leq t \leq 1.1$ s. The resulting trajectory is shown in Fig. 4. The evaluated costs of the trajectories resulting from SPOCP-trained policy with and without the applied disturbance are given in Tab. 1.

Table 1 Costs of Nominal SPOCP trajectories

Trajectory	Cost [N^2 s]
Policy (nominal)	353.0
Policy (disturbed)	395.2
Optimal	338.0

The policy trained solely on SPOCP trajectories has demonstrated not just an ability to perform near-optimal feedback control to a landing point, but that it can recover from a disturbance that puts it in an off-nominal trajectory.

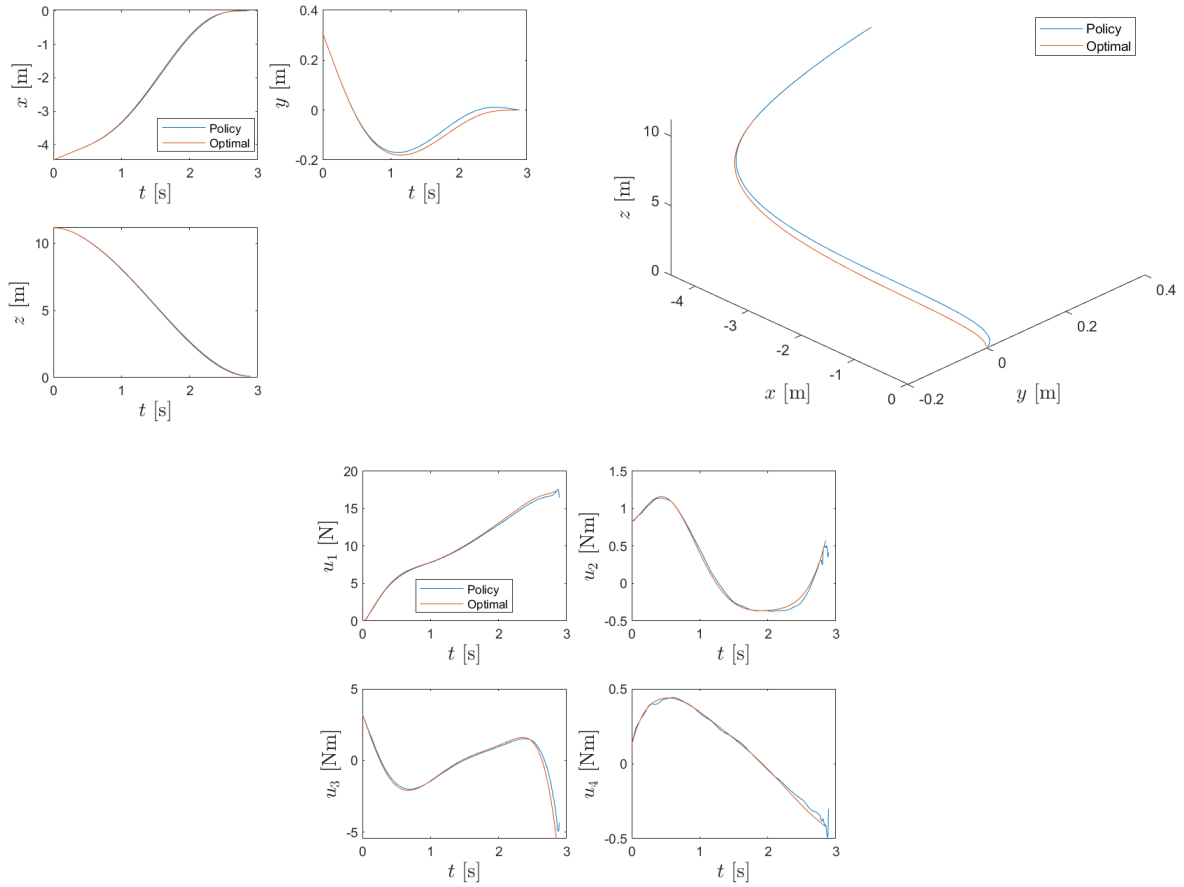

Fig. 3 Trajectory and control history for policy trained on SPOCP

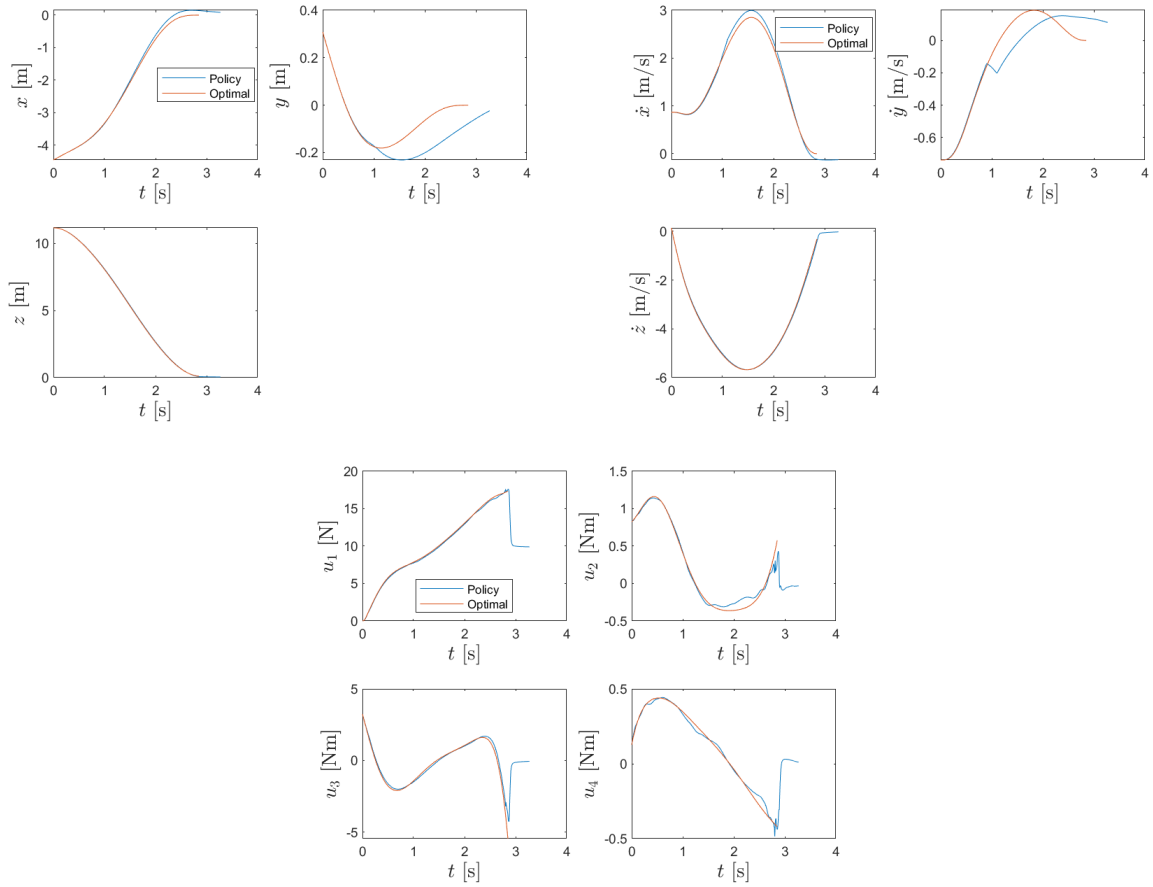

Fig. 4 Trajectory and control history for disturbed policy trained on SPOCP trajectories

While this demonstration alone is not an analytical proof on the closed-loop stability of the trained policy, it does represent an important achievement in producing a robust controller that can recover to its target landing point despite the presense of a disturbance. The next two sections detail how the policy was retrained using disturbed open-loop optimal trajectories to mitigate the loss of optimality in the presence of a disturbance.

B. Policy trained on MPOCP

After incorporating MPOCP trajectories into the training data, the policy was trained and simulated with the applied disturbance. The resulting policy-driven trajectory along with the corresponding MPOCP trajectory is shown in Fig. 5. The evaluated costs of the trajectories are shown in Tab. 2.

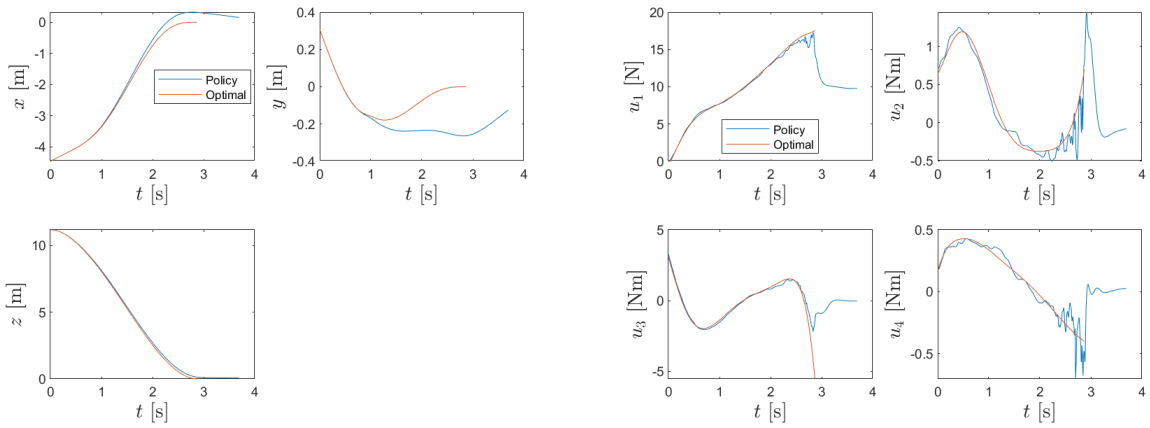

Fig. 5 Trajectory and control history for policy trained on MPOCP trajectories

Table 2 Costs of MPOCP trajectories

Trajectory	Cost [N^2 s]
Policy (disturbed)	431.2
Optimal	343.6

The policy trained on MPOCP trajectories demonstrated a loss in performance by a few metrics. First, from visual inspection of Fig. 5, the policy does not demonstrate complete recovery to the landing site. While it does recover the x and z coordinates, the y coordinate does not recover completely. Second, the evaluated cost of the policy-driven trajectory is 25.5% higher than optimal as seen in Tab. 2. This demonstration represents a challenge in imitation learning for multi-phase feedback optimal control.

C. Policy Trained on 3SPOCP

After incorporating 3SPOCP trajectories into the training data, the policy was trained and simulated with the applied disturbance. The resulting policy-driven trajectory along with the corresponding MPOCP trajectory is shown in Fig. 6. The evaluated costs of the trajectories are shown in Tab. 3.

Table 3 Costs of 3SPOCP trajectories

Trajectory	Cost [N^2 s]
Policy (disturbed)	374.9
Optimal	354.3

The policy trained on 3SPOCP trajectories demonstrated a gain in performance. First, the cost of the disturbed policy-driven trajectory is 5.1% less than the cost of the trajectory driven by the policy trained solely on SPOCP

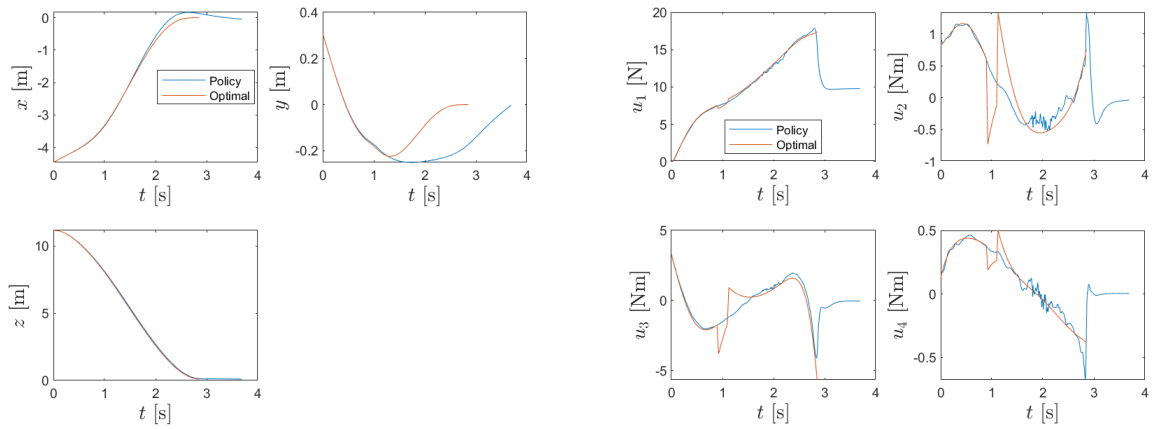

Fig. 6 Trajectory and control history for policy trained on 3SPOCP trajectories

trajectories. In addition, this cost is only 5.8% more than the corresponding 3SPOCP optimal trajectory. The reduction in cost that resulted in including 3SPOCP trajectories in the training data represents a new capability to mitigate loss of optimality when disturbances are present.

The difference in performance between the policy trained on MPOCP trajectories and the policy trained on 3SPOCP trajectories should be noted. The former is optimized over the entire trajectory while the latter is optimized over each "phase" individually. By optimizing over the entire trajectory for MPOCP, a successfully trained policy would represent one that effectively knows that disturbance is to take place at a specific time-interval before it occurs. The entire trajectory then becomes off-nominal (even before the disturbance is applied), as opposed to just the part of the trajectory during and after the applied disturbance. Extracting a distribution of states for training data that accurately matches the distribution of states seen in roll-out then becomes challenging when the goal is to train the policy on MPOCP trajectories. When optimizing over each phase individually for 3SPOCP, the trajectory only becomes off-nominal once the disturbance is applied, and the trained policy represents one that does not know about the disturbance until it drives the lander into the distribution of states seen only by disturbed trajectories. It is therefore less challenging to extract a distribution of states for training data that matches the distribution of states seen in rollout, since the disturbance is applied in a similar way. Instead of the disturbance affecting the trajectory before its application (like in the MPOCP case), the 3SPOCP trajectory is only affected after the disturbance is applied.

VI. Conclusion

This paper showed the ability of DNNs to learn from open-loop optimal trajectories and perform near-optimal feedback control in a robust way. The optimal landing problem was cast to include the presence of constant disturbances in defined time-windows. The use of multi-phase optimal control and single-phase optimal control were both investigated. The policy showed a greater challenge in learning from MPOCP trajectories than from 3SPOCP trajectories. However, it was shown that the policy can learn to mitigate loss of optimality in the presence of disturbance by including open-loop trajectories from disturbed optimal landings in the training set.

While it was shown that given a relatively small amount of training data, a DNN with a simple architecture can learn to perform near-optimal, feedback control of the 6DOF lander, this controller does not maintain near-optimality through an applied disturbance during a finite time window. This study reconciled this loss of optimality by generating large sets of multi-phase optimal trajectories and triple single-phase optimal control problems that correspond to control histories that maintain optimality through the disturbance. Imitation learning techniques then used these large sets of optimal examples with disturbances to train a policy that performed stable, near-optimal feedback control of a lander in the presence of disturbances.

Future works should further explore the space of possible disturbances. In this study a constant disturbance was applied in a finite time-window. Time-varying disturbances are also present in real applications, so they should also be investigated. In addition, future works include analytical assessments and proofs on the stability of the the trained policy. So far, stability has primarily been shown via demonstration, which does not allow for strong claims and assessments on

stability. A rigorous, mathematical proof that shows that the policy is stable in the sense of Lyapunov would allow for such a claim.

The previous works discussed in Section III.C have shown that DNNs can learn near-optimal feedback control from numerically generated optimal examples for low-fidelity formulations like the 3DOF lander. Disturbances are present in many optimal control applications, not just the pinpoint landing problem. Where appropriate, disturbances can be modeled as either multi-phase optimal control problems to provide examples for policies to learn how to maintain optimality in off-nominal trajectories where the disturbance is known, or as triple single-phase optimal control problems to provide examples where the disturbance is not known. A library of disturbed, optimal trajectories can be built and used to train the DNN, thus providing a way to train policies to not only perform near-optimal feedback control in nominal trajectories, but to also mitigate loss of optimality when disturbances are present.

Acknowledgments

This investigation was supported by the NASA Space Technology Graduate Research Opportunity (Grant Number 80NSSC20K1188).

References

[1] Myers, S. L., and Chang, K., "China lands chang'e-5 spacecraft on Moon to gather lunar rocks and soil," *The New York Times*, 2020. URL https://www.nytimes.com/2020/12/01/science/china-moon-landing.html.

[2] Harwood, W., "NASA says first Artemis Moon landing now expected in 2025," , Nov 2021. URL https://www.cbsnews.com/news/nasa-says-first-artemis-moon-landing-slipping-to-2025/.

[3] Sengupta, S., "India launches Unmanned Orbiter to moon," *The New York Times*, 2008. URL https://www.nytimes.com/2008/10/22/world/asia/22indiamoon.html.

[4] Krizhevsky, A., Sutskever, I., and Hinton, G. E., "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, Vol. 25, 2012.

[5] Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A.-r., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T. N., et al., "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal processing magazine*, Vol. 29, No. 6, 2012, pp. 82–97.

[6] Sutton, B. A. G., Richard S., *Reinforcement Learning: An Introduction*, 2nd ed., A Bradford Book, Cambridge, Massachusetts, 2014.

[7] Betts, J. T., *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming*, 2nd ed., Cambridge University Press, USA, 2009.

[8] Tailor, D., and Izzo, D., "Learning the optimal state-feedback via supervised imitation learning," *Astrodynamics*, Vol. 3, No. 4, 2019, pp. 361–374.

[9] Furfaro, R., Bloise, I., Orlandelli, M., Di Lizia, P., Topputo, F., Linares, R., et al., "Deep learning for autonomous lunar landing," *2018 AAS/AIAA Astrodynamics Specialist Conference*, Vol. 167, Univelt, 2018, pp. 3285–3306.

[10] Pomerleau, D. A., "Alvinn: An autonomous land vehicle in a neural network," *Advances in neural information processing systems*, Vol. 1, 1988.

[11] Ross, S., Gordon, G., and Bagnell, D., "A reduction of imitation learning and structured prediction to no-regret online learning," *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, JMLR Workshop and Conference Proceedings, 2011, pp. 627–635.

[12] Bishop, C. M., "Probability Distributions," *Pattern Recognition and Machine Learning*, Springer, 2006, pp. 67–136.

[13] Halbert, D. C., "Programming by example," 1984.

[14] Bojarski, M., Testa, D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., Jackel, L., Monfort, M., Muller, U., Zhang, J., Zhang, X., Zhao, J., and Zieba, K., "End to End Learning for Self-Driving Cars," 2016.

[15] Kober, J., and Peters, J., "Policy Search for Motor Primitives in Robotics," *Mach. Learn. J*, Vol. 84, 2008, pp. 171–203. https://doi.org/10.1007/s10994-010-5223-6.

[16] Laskey, M., Lee, J. N., Fox, R., Dragan, A. D., and Goldberg, K., "DART: Noise Injection for Robust Imitation Learning," *CoRL*, 2017.

[17] MATLAB, *version 7.10.0 (R2010a)*, The MathWorks Inc., Natick, Massachusetts, 2010.

[18] Koenemann, J., Licitra, G., Alp, M., and Diehl, M., "OpenOCL - Open Optimal Control Library," 2019.

[19] Wächter, A., and Biegler, L. T., "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical Programming*, Vol. 106, 2006, pp. 25–57.

[20] Sánchez-Sánchez, C., Izzo, D., and Hennes, D., "Learning the optimal state-feedback using deep networks," *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, 2016, pp. 1–8.

[21] Sánchez-Sánchez, C., and Izzo, D., "Real-time optimal control via Deep Neural Networks: study on landing problems," *ArXiv*, Vol. abs/1610.08668, 2016.

[22] Gaudet, B., Linares, R., and Furfaro, R., "Deep Reinforcement Learning for Six Degree-of-Freedom Planetary Landing," *Advances in Space Research*, Vol. 65, 2020. https://doi.org/10.1016/j.asr.2019.12.030.

[23] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X., "TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems," , 2015. URL https://www.tensorflow.org/, software available from tensorflow.org.

[24] Kingma, D. P., and Ba, J., "Adam: A Method for Stochastic Optimization," *CoRR*, Vol. abs/1412.6980, 2015.