Solving the Soft Landing Problem with the Implicit Actor Critic Framework

Julian Wang
Virginia Tech
Blacksburg, VA 24061
julianw@vt.edu

Karlton Wirsing
Virginia Tech
Manassas, VA 20109
kwirsing@vt.edu

Adrian Ruvalcaba
Virginia Tech
Blacksburg, VA 24061
adrianr@vt.edu

Abstract

The soft landing of a space vehicle on a celestial object has been a problem studied since the beginning of the space race. Since then, the soft landing problem has been studied to autonomously land rovers on the surface of Mars and to recover rocket stages after launch. As new technologies and materials allow for even bolder space exploration missions, there is a need for more robust solutions to the soft landing problem to meet the challenge. This paper will augment the lossless convexitifed soft landing optimal control problem with the implicit critic framework to improve the solution's robustness from uncertain environmental factors.

1 Introduction

The past two decades have seen the emergence of "NewSpace" ventures. These privately backed space ventures have been created to tackle aerospace problems, deemed too high risk by national space agencies, for the sake of idealism and for the pursuit of profit. The one thing that unites all of these NewSpace ventures is the assumption that access to space will become democratized. One of the most prominent NewSpace ventures, SpaceX, has sought to decrease the cost to access space by streamlining manufacturing and making its rockets reusable.

Autonomous landing requires sufficient knowledge of the vehicle model and environment to achieve a successful soft landing. With sufficient knowledge, a soft landing descent can be formulated and solved as a convex optimization problem. However, knowledge of the vehicle model and environment will never be perfect. Therefore, we need a critic to assess an actor's performance over time and to provide feedback. In this paper, we will solve the soft landing problem with an implicit actor-critic (iAC) model [iAC21]. The actor will be the optimal solution to the soft-landing problem. We introduced measurement and model uncertainties at each discrete timestep. The critic's job is to minimize the solution error due to noise over time.

2 Soft Landing Problem

The soft-landing optimal control problem can be split into 2 problems: the Minimum Landing Error (MLE) problem and the Minimum Fuel (MF) problem [AP07]. The MLE problem states the final position of the lander after the landing descent should be as close to the target position as possible. The MF problem states the total fuel consumed should be minimized in order to find the most fuel efficiency landing descent. The two problems are solved in series to simultaneously minimize the landing error and the amount of fuel consumed. The objective function for MLE problem and MF problem are described in 1 and 2 respectively.

$$\min_{t_f, \mathbf{T}_c} \|\mathbf{r}(t_f) - \mathbf{q}\| \qquad (1)$$

$$\min_{t_f, \mathbf{T}_c(\cdot))} \int_0^{t_f} \alpha \|\mathbf{T}_c(t)\| \, dt \qquad (2)$$

Where t is the time since the soft landing operation has begun, t_f is the final time in the time domain, r is the lander's position at time t, the q is the target final position, $\mathbf{T}_c$ is the lander's thrust vector at time t, and α is the propellant mass depletion rate. The MLE problem's objective function is fairly straightforward. We want to minimize the error between the final lander position and the target position. The MF's objective function is a bit more roundabout. By minimizing the thrust magnitude over time, we will in effect minimize the total fuel consumption as well.

2.1 Soft Landing Constraints

The constraints of the MLE and MF problems can be categorized into boundary value, vehicle dynamics, and propulsion constraints. The boundary value constraints fix the initial and final values of our problem. The vehicle dynamic constraints anchor the trajectory to the physics of the environment. And finally, the propulsion constraints limits the possible actions from the spacecraft due to the design of the rocket engine. Starting with the boundary value constraints, the initial and final value constraints are described in 3 and 4.

$$\mathbf{r}(0) = \mathbf{r_0}, \;\; \dot{\mathbf{r}}(0) = \dot{\mathbf{r}}_0, \;\; m(0) = m_0 \quad (3) \qquad \dot{\mathbf{r}}(t_f) = 0, \;\; m(t_f) \geq m_f > 0 \quad (4)$$

Where $m(t)$ is the mass of the vehicle at time t, $\mathbf{r_0}$ is the initial lander position, $\dot{\mathbf{r}}_\mathbf{0}$ is the initial lander velocity, m_0 is the initial lander mass, and m_f is the lander's dry mass. The initial value constraints are provided by the initial conditions of the problem. The lander will begin its soft landing descent with a known position, velocity, and mass. On the other hand, the final value constraints are defined by the problem. We want the final velocity of the lander to be zero in order to achieve a soft landing. And we want the lander mass time history to be always greater than the lander's dry mass.

The vehicle dynamic constraints are defined by equation of motions of the lander. For simplicity, we will assume a rigid lump mass vehicle landing on a rotating planetary object. The planetary object will not have sufficient atmosphere for aerodynamic forces during terminal descent. An example would be the Moon or Mars. The vehicle kinematics and mass rate are described in Eq. 5 and 6.

$$\dot{\mathbf{x}} = \mathbf{A}(\omega)\mathbf{x}(t) + \mathbf{B}\left(\mathbf{g} + \frac{\mathbf{T}_c(t)}{m(t)}\right) \quad (5) \qquad \dot{m}(t) = -\alpha \left\|\mathbf{T}_c(t)\right\| \quad (6)$$

Where $\mathbf{x}$ is the lander state (position and velocity) vector at time t, $\dot{\mathbf{x}}$ is the lander state derivative (velocity and acceleration) vector at time t, $\mathbf{g}$ is the planetary object's constant gravity vector, and ω is the planetary object's constant angular velocity vector. The kinematic equation describes the expected change in the lander's state over time and due to actions exerted by the lander. Since thrust is required for a soft landing, the mass of the vehicle will change over time as propellant is expelled. The acceleration forces $\mathbf{g}$ and $\mathbf{T}_c(t)$ affect a change in the vehicle's velocity, the matrix $\mathbf{B}$ maps the acceleration forces to the velocity component of the state vector. The angular velocity vector ω is used to calculate $\mathbf{A}$. $\mathbf{A}$ is the rotation matrix that transforms the vehicle states from the planet's rotating frame to the planet's fixed frame. The rotation matrix $\mathbf{A}$ is defined in Eq. 7 and Eq. 8.

$$\mathbf{A}(\omega) = \begin{bmatrix} 0 & \mathbf{I} \\ -\mathbf{S}(\omega)^2 & -\mathbf{S}(\omega) \end{bmatrix} \quad (7) \qquad \mathbf{S}(\omega) = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix} \quad (8)$$

The propulsion constraints are defined by the limitations of the rocket engine. The lander achieves thrust vectoring by gimballing the rocket engine and by adjusting the rocket's thrust level. However, there are control constraints on the thrust vectoring control. The rocket engine cannot produce thrust beyond its design limit. Nor can the rocket engine produce thrust below a certain threshold for the engine to be self-sustaining. Finally, the thrust vector cannot exceed the gimbal's angle limit. The three control constraints can be written into Eq. 9 and 10.

$$0 < \rho_1 \leq \left\|\mathbf{T}_c(t)\right\| \leq \rho_2 \quad (9) \qquad \hat{\mathbf{n}}^T \frac{\mathbf{T}_c(t)}{\left\|\mathbf{T}_c(t)\right\|} \geq \cos\theta \quad (10)$$

Where ρ_1 is the lower thrust bound, ρ_2 is the upper thrust bound, $\frac{\mathbf{T}_c(t)}{\|\mathbf{T}_c(t)\|}$ is the thrust unit vector in the fixed frame, $\hat{n}$ is the unit direction vector of the lander in the fixed frame, and θ is the maximum allowable angle between the lander's unit direction vector and the thrust unit vector.

The MLE problem and the MF problem inherits the same boundary value, vehicle dynamics, and propulsion constraints. However, the MF problem does have an additional constraint. The solution to the MLE error problem to used as a constraint to the minimum fuel problem. The MF's additional landing error constraint is a revised version of the MLE's objective function. It is described in Eq. 11.

$$\|\mathbf{r}(t_f) - \mathbf{q}\| \leq \|\mathbf{d}^*_{MLE} - \mathbf{q}\| \tag{11}$$

Where $\mathbf{d}^*_{MLE}$ is the final lander position from the optimal solution to the MLE problem. Basically, we don't want the final landing error from the MF problem to be worse than the MLE. Otherwise, it would defeat the whole purpose of trying to land as closely to the target position.

The MLE and MF problems can be constructed using the constraints discussed in this section. However, they cannot be solved using convex optimization because the vehicle dynamics and propulsion constraints. The vehicle dynamic constraint is nonlinear because of $\frac{\mathbf{T}_c(t)}{m(t)}$. And the propulsion constraint is nonconvex because of the lower thrust boundary constraint.

2.2 Constraint Convexification and Linearization

The nonconvex constraints can be convexified through the introduction of several relaxation constraint variables [MRS$^+$21]. Starting with the nonconvex propulsion constraints, we can introduce a slack variable Γ to relax the thrust magnitude $\leq \|\mathbf{T}_c(t)\|$. This will make the lower thrust constraint convex. We can then introduce three additional variables to linearize the vehicle dynamic constraints. We will introduce the following change of variables in Eq. 12.

$$\sigma \triangleq \frac{\mathbf{\Gamma}}{m}, \qquad \mathbf{u} \triangleq \frac{\mathbf{T}_c}{m}, \qquad z \triangleq \ln m, \tag{12}$$

The introduction of substitution variables make the propulsion constraints nonconvex again. We will need to apply a second order cone approximation to the changed thrust vector bounds [AP07].

2.3 Convexified Linearized Soft Landing Problem Statement

With all of the objective functions defined and the constraints convexified and linearized, we can finally define the convexified MLE and MF problems so it can be solved using convex optimization techniques. The two problems are defined in full below.

Convexified Linearized Minimum Landing Error Problem:

$$\min_{t_f, \mathbf{u}} \|\mathbf{r}(t_f) - \mathbf{q}\| \tag{13}$$

$$\text{s.t.} \quad \mathbf{r}(0) = \mathbf{r_0}, \quad \dot{\mathbf{r}}(0) = \dot{\mathbf{r}}_0, \quad z(0) = \ln m_0 \quad \dot{\mathbf{r}}(t_f) = 0, \quad z(t_f) \geq \ln m_f > 0 \tag{14}$$

$$\dot{\mathbf{x}} = \mathbf{A}(\omega)\mathbf{x}(t) + \mathbf{B}(\mathbf{g} + \mathbf{u}(t)) \qquad \dot{z}(t) = -\alpha\sigma(t) \tag{15}$$

$$\rho_1 e^{-z_0}[1 - (z(t) - z_0(t)) + \frac{(z(t) - z_0(t))^2}{2}] \leq \sigma(t) \leq \rho_2 e^{-z_0}[1 - (z(t) - z_0(t))] \tag{16}$$

$$\|\mathbf{u}(t)\| \leq \sigma(t), \qquad \hat{\mathbf{n}}^T \mathbf{u}(t) \geq \cos\theta \tag{17}$$

Convexified Linearized Minimum Fuel Problem:

$$\min_{t_f, \sigma} \int_0^{t_f} \alpha \, \|\mathbf{u}(t)\| \, dt \tag{18}$$

s.t. the constraints defined by Eq. 14 - 17 and:

$$\|\mathbf{r}(t_f) - \mathbf{q}\| \leq \|\mathbf{d}^*_{MLE} - \mathbf{q}\| \tag{19}$$

3 Implicit Actor Critic Framework

A spacecraft performing a soft landing descent in a real environment will encounter unknown uncertainties. Since the solution to the soft-landing problem is based on an idealized model and environment, its predicted actions will be sub-optimal. Therefore, we need a robust solution which minimizes the actor's model and measurement uncertainty in order to improve policy decision making. In this section, we will explore how an Implicit Actor-Critic framework can be utilized to optimize the soft landing problem by evaluating the value of its current and future actions [iAC21].

3.1 Actor Optimization Model

As it was previously mentioned, in real life, the actor will create sub-optimal actions due to uncertainties. The uncertainties may be imperfect knowledge of the state model or the physics of the environment. Therefore, uncertainties will cause the actor to predict noisy future states. The noisy future states will lead to sub-optimal policy mapping and actions.

In order to correct for uncertain knowledge, we will have a critic which monitors the actor's past performance. The actor will send a batch of actions and its states from $k - n$ to k to be evaluated by the critic. Therefore, the actor needs to perform the n steps to collect data for the critic to process. The critic should be able to correct the actor's parameters which were perturbed due to uncertainties. In this paper, we will only assume uncertain position knowledge to simplify the problem. In reality, the actor's parameters should include the vehicle's velocity and mass.

3.2 Critic Optimization Model

The critic contains a noise-free model of the actor. It will use its noise-free model to predict the ideal past states based on the previous actions of the actor. The real and ideal states will be used to calculate the real and ideal cost. The real cost reflects the performance of the actor due to uncertainties. The ideal cost reflects the performance of the actor if there was zero uncertainty. The cost function computes the total cost for the batch real and ideal states. Ideally, the cost function would incorporate the vehicle's velocity and mass. The implemented cost function is described in 20.

$$c_k = w \sum_{i=1}^{n} \tag{20}$$

In order to estimate Q, we will using a Reward Warping Function (RWF) to map the ideal costs to the real costs. The implemented RWF is described in Eq. 21

$$Q_{est} = \mathrm{RWF}(c, \omega) = w \sum_{i=1}^{n} c_{i,i} \tag{21}$$

Where n is the size of the batch and $c_{i,i}$ is the ideal cost at batch index i. The critic parameter ω is a scaling factor used to fit the real costs to the ideal costs.

3.3 Critic and Actor Update

At the start of the soft landing descent, the critic parameter ω is seeded with some initial value. Over time, it is updated based on the difference between the estimated Qs and the real costs. We will update ω using a least squares minimization problem outlined in 22. Over time as ω gets updated, the RWF will map the ideal costs closer to the real costs.

$$\min_{\omega} \|c_{r,k} - Q(\omega)\|_2^2 . \tag{22}$$

Where $c_{r,k}$ is the real cost at step k. Finally, we can correct the actor's parameter ζ by calculating the gradient of the estimated Q with respect to the actor's parameter ζ. Ideally we would deploy a Python library such as CVXPYLayers to commute the Q gradient [AAB$^+$19]. But due to time constraints, we calculated the Q gradient as a simple slope between Q and the states in the batch.

$$\nabla_\zeta Q_k = \frac{Q_k - Q_{k-n}}{s_k - s_{k-n}} \tag{23}$$

The estimated Q gradient is used to update ζ via a gradient ascent described in 24. The estimated Q gradient is augmented with a rate learning α.

$$\zeta_k \longleftarrow \zeta_k + \alpha \nabla_\zeta Q_{k+1} \tag{24}$$

4 Results

We will solve the soft landing problem in an idealized Martian environment. We will use the initial conditions outlined in the numerical example of Ackmese et al. [ACB13]. The actor was perturbed with a Gaussian noisy position vector to simulate uncertain position knowledge. The noisy position vector had a mean of 0 and a standard deviation of 1 meters. A seed of the noisy position vector was generated to ensure repeatability from run to run. The critic was designed with state, action, cost, and Q estimate buffer size of 10. The initial critic parameter value was assumed to be $\omega = 1$. The learning rate of the actor parameter was assumed to be $\alpha = 1\mathrm{e}-4$.

We solved the minimum landing error problem with the iAC framework and with just the actor. Both cases were seeded with the noisy position over time to ensure comparability. For our experiments, we found the critic was able to provide meaningful feedback to the actor. When the actor was alone, we had a final landing error of 0.75073 meters with a final mass of 1734.79 kg. When the critic was guiding the actor, we had a final landing error of 0.7477 meters and a final mass of 1734.79 kg. While the final lander mass is the same, the landing error is slightly better.

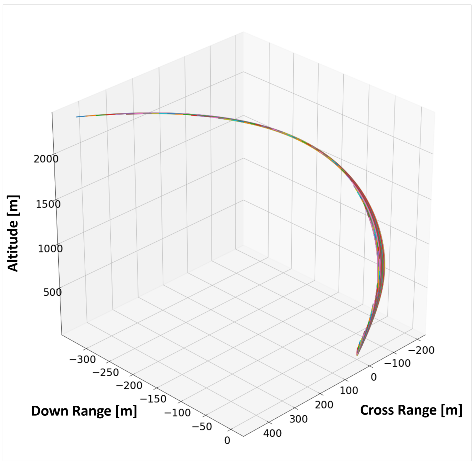

Figure 1: Minimum Landing Error Solution Over Time w/ IAC

5 Conclusion

In this paper, we have demonstrated that the soft landing problem can be convexified and linearized to be solved using a convex optimization tool such as CVXPY. We introduced a lander which would use the convexified soft landing problem to achieve a soft landing on Mars in discrete time. We have designed and implemented an implicit actor critic framework to guide the lander in a noisy environment. We demonstrated in a noisy environment, the critic was able to guide the lander slightly closely to the target.

However in this paper, we considered the actor's parameter to only be the vehicle's position, when it should also include the vehicle's velocity and mass. The calculation of the Q estimate gradient was crude. We should have used a tool like CVXPYLayers to perform the calculations. Finally, the addition of model noise to the convex problem is a bit uncertain and should be re-examined in the future.

6 Acknowledgements

Julian Wang - Abstract, Introduction, Section 2, Section 3, Section 4, Section 5

Karlton Wirsing - Section 3.2, Initial Critic Implementation, General Code Improvements

Adrian Ruvalcaba - Section 3.1, Initial Critic Implementation

References

[AAB+19] A. Agrawal, B. Amos, S. Barratt, S. Boyd, S. Diamond, and Z. Kolter. Differentiable convex optimization layers. In *Advances in Neural Information Processing Systems*, 2019.

[ACB13] Behçet Açıkmeşe, John M Carson, and Lars Blackmore. Lossless convexification of nonconvex control bound and pointing constraints of the soft landing optimal control problem. *IEEE Transactions on Control Systems Technology*, 21(6):2104–2113, 2013.

[AP07] Behcet Acikmese and Scott R Ploen. Convex programming approach to powered descent guidance for mars landing. *Journal of Guidance, Control, and Dynamics*, 30(5):1353–1366, 2007.

[iAC21] Actor-critic reinforcement learning in continuous state-action spaces with optimization as deterministic policies. 2021.

[MRS+21] Danylo Malyuta, Taylor P Reynolds, Michael Szmuk, Thomas Lew, Riccardo Bonalli, Marco Pavone, and Behcet Acikmese. Convex optimization for trajectory generation. *arXiv preprint arXiv:2106.09125*, 2021.