doi: 10.82124/CEAS-GNC-2026-007 CEAS-GNC-2026-007

EuroGNC Navigation Guidance Control CEAS Madrid, Spain May 5th-7th uc3m Universidad Carlos III de Madrid AIAA 2026

Deep Reinforcement Learning and Optimal Control for Designing Safe Orbits around Asteroids

Julio C. Sanchez Associate Professor, Universidad de Sevilla, Sevilla, Spain. jsanchezm@us.es

Pablo Redondo-Amaro MSc Student, Universidad de Sevilla Sevilla, Spain. pabredama@alum.us.es

ABSTRACT

The inhomogeneous gravity field of an asteroid strongly perturbs nearby spacecraft trajectories. Orbits that would be bounded under Keplerian assumptions may instead evolve into collision or escape paths. This work addresses the challenge of designing safe and optimal control strategies for low-altitude orbits around asteroids. Two solution approaches are compared: direct transcription of the optimal control problem, and the soft actor-critic algorithm, a deep reinforcement learning method. Direct transcription boils down to a non-linear program which proves to be efficient in terms of delta-v consumption but becomes infeasible for certain test cases. In contrast, the soft actor-critic method produces a global policy that succeeds across the entire test set and offers faster evaluation. In both approaches, an additional penalty term is introduced to keep the spacecraft trajectory within a safe orbital shell.

Keywords: Asteroids, Astrodynamics, Reinforcement Learning, Trajectory Optimization

Nomenclature

a, b, c	=	ellipsoid principal axes
$\mathbf{a}_{\text{grav}}$	=	gravity acceleration
$\mathcal{D}$	=	replay buffer
J	=	objective function
$\mathcal{H}$	=	entropy
$\mathcal{L}$	=	loss function
Q	=	action-value function
$\mathbf{r}$	=	spacecraft position
$\mathbf{x}$	=	spacecraft state
$\mathbf{v}$	=	spacecraft velocity
$\Delta\mathbf{v}$	=	spacecraft impulse
t	=	time
π	=	actor policy
$\boldsymbol{\omega}$	=	asteroid angular velocity
μ	=	standard gravity parameter

1 Introduction

Asteroids present a fascinating but challenging environment for orbital dynamics because their small size, irregular shapes, and heterogeneous internal structure create a highly inhomogeneous gravity field. Unlike planets or large moons, which can often be approximated by smooth, nearly spherical gravitational potentials, asteroids exhibit significant local variations in mass distribution that distort the surrounding gravitational field. As a result, orbits around asteroids are often unstable since the gravity perturbation can quickly grow causing spacecraft to drift, impact the surface, or escape entirely [1]. Accordingly, careful trajectory prediction and corrective control actions are key to avoid a catastrophic mission loss.

In the pioneer NEAR mission to asteroid Eros, orbital maneuver design relied on ground assets and frequent communication using the Deep Space Network [2]. The last operational orbit was placed far enough from the surface to be stable but still required station-keeping maneuvering. During that last phase NEAR also made two low altitude flybys which were designed taking into account Eros estimated gravity field. OSIRIS-REx mission to asteroid Bennu broke the orbit record as the closest ever to a celestial body surface with an operational altitude of 680 meters [3]. The difference is that Bennu gravity field is extremely weak thus OSIRIS-REx largely exploited hyperbolic arcs due to a cheap delta-v. In terms of on-board autonomy NEAR did not incorporate any means while OSIRIS-REx kept it limited by triggering orbital maneuvers at specified checkpoints. Mastering on-board autonomy for small bodies could potentially allow to perform high risk-high reward operations even if the dynamics are not as reactive as Bennu. Relying on ground assets for that type of operation is usually infeasible due to communication delays at astronomical distances and insufficient accessibility to the Deep Space Network. Accordingly, this work focuses on the problem of safely orbiting unstable regions around asteroids without jeopardizing the well-posed behavior in the stable domain. In that sense, the technical goal is to design safe controlled orbits with a minimal delta-v consumption.

A rotating gravity field presents equilibrium points and the existence of periodic orbits [4]. These dynamical structures could be exploited as it is the case of hovering around a point in the asteroid-centred body-fixed frame [5]. Other works [6–9] have focused on extending the safe region to the orbital domain. Reference [6] integrated model predictive control with gravity estimation to a controlled orbit confined in a spherical surface. However, control cost is large since the semimajor axis perturbations is always counteracted. The manuscript [7] proposes a path-following control with sliding surfaces to induce artificial controlled orbits. This method also requires a persistent continuous active control to counteract perturbations. The work [8] uses reinforcement learning, in particular the soft actor-critic algorithm [10], to control orbits by using different gravity models. The way rewards are designed is convenient since the spacecraft path could be any as long as it does not collide or escape. However, the reported number of collisions for the best controlled scenario is seemingly high ($\approx 50\%$). Lastly, reference [9] proposes an event-based controller by only acting when the spacecraft is predicted to abandon a safe orbital shell. This seems convenient for the sake of avoiding the frequent repetition of impulses.

In this work, the safe orbiting problem around the asteroid is posed as a constrained optimal control problem in terms of delta-v. Besides collision and escape constraints, the spacecraft is incentivized to remain within a spherical orbital shell by using a penalty cost. The problem is solved via two different methods: direct transcription of the optimal control problem (OCP) and the soft-actor critic (SAC) which is a deep reinforcement learning algorithm. The first method, direct transcription, discretizes the original optimal control problem transforming it into a static non-linear program. The soft-actor critic is based on the exploration and exploitation concept to approximate a Q-network which serves to guide a control policy toward actions that maximize both expected reward and entropy. The goal is to compare these approaches in terms of safety, fuel consumption and computational burden. Direct transcription of the optimal control problem yields lesser fuel consumption than the soft-actor critic. However, the soft-actor critic tends to succeed in safety due to exploration and has a lower computational burden after the agent is trained.

This paper is structured as follows: Section 2 presents asteroid dynamics; Section 3 states the optimal control problem and the orbital shell penalty cost; Section 4 describes the direct transcription method for optimal control and the soft-actor critic; Section 5 shows numerical simulations of interest and compares both approaches; Section 6 provides conclusions and future work to be explored.

2 Dynamics around an asteroid

In this section, the spacecraft equations of motion are derived in the asteroid-centred body-fixed frame. Subsequently, the asteroid gravity field and its shape model are presented.

2.1 Spacecraft equations of motion

Let us define the asteroid-centred inertial frame as N and the asteroid-centred body-fixed frame as A. Both of these frames have their origin at the asteroid center of mass. The asteroid-centred body-fixed frame is rigidly attached to the asteroid and rotates with respect to the inertial frame. The rotational motion of the majority of asteroids can be well approximated by a constant rotation around the major inertia axis. Let us define z_A as the major inertia axis and the plane $x_A y_A$ as the asteroid equatorial plane. Consequently, the angular velocity of frame A with respect to N can be expressed as $\boldsymbol{\omega}_{A/N} = \omega \mathbf{k}_A \equiv$ constant. For the sake of simplicity $z_N \equiv z_A$ is chosen.

In this work, the spacecraft state is represented in the A frame. This follows two reasons: 1) The asteroid shape is invariant in the A frame thus the collision constraint does not change with respect time; 2) Spacecraft equations of motion are autonomous in the A frame since the asteroid gravity field is static in this frame. Accordingly, the spacecraft equations of motion expressed in the A frame are

$$\begin{aligned} \dot{\mathbf{r}} &= \mathbf{v} \\ \dot{\mathbf{v}} &= -\boldsymbol{\omega} \times (\boldsymbol{\omega} \times \mathbf{r}) - 2\boldsymbol{\omega} \times \mathbf{v} + \mathbf{a}_{\text{grav}} \end{aligned} \tag{1}$$

where the spacecraft position vector is $\mathbf{r} = [x, y, z]^T$, the spacecraft velocity vector is $\mathbf{v} = [v_x, v_y, v_z]^T$ and $\mathbf{a}_{\text{grav}}$ is the asteroid gravity which is the single dynamical effect to be taken into account. Note that $\boldsymbol{\omega} \equiv \boldsymbol{\omega}_{A/N} = [0, 0, \omega]^T$. In this work, only impulsive control is assumed thus when thrusters are fired at a discrete instant t_k, the spacecraft velocity vector varies instantaneously as

$$\mathbf{v}(t_k^+) = \mathbf{v}(t_k^-) + \Delta\mathbf{v}_k \tag{2}$$

where $\Delta\mathbf{v}_k$ is the applied impulse vector.

2.2 Asteroid gravity field

Asteroids typically have irregular shapes and heterogeneous internal density distributions. For asteroids where gravity dominates over solar radiation pressure, the gravity field becomes the primary source of perturbation affecting a spacecraft in low orbit. Several models exist in the literature to represent an asteroid's gravity field, including spherical harmonics, polyhedron models, mascon approaches, and more recent data-driven formulations. Among these, spherical harmonics are commonly used in astrodynamics; however, for highly irregular or elongated bodies, this representation diverges within the circumscribing sphere of the asteroid. Consequently, in low-altitude regimes around such asteroids, alternative gravity models—such as polyhedron or mascon representations—are generally preferred.

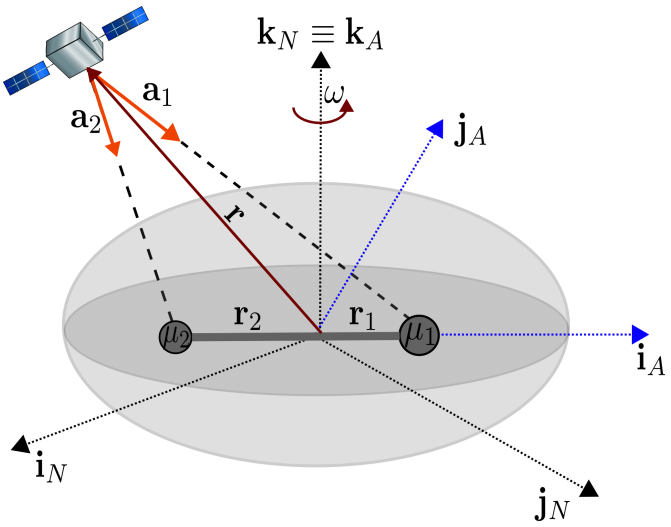

Fig. 1 Asteroid model and frames of reference.

In this work, for the sake of simplicity, the asteroid gravity is modeled by two masses attached with a massless rod [11]. Consequently, the gravity expression is

$$\mathbf{a}_{\text{grav}} = -\mu_1 \frac{\mathbf{r} - \mathbf{r}_1}{\|\mathbf{r} - \mathbf{r}_1\|^3} - \mu_2 \frac{\mathbf{r} - \mathbf{r}_2}{\|\mathbf{r} - \mathbf{r}_2\|^3} \tag{3}$$

It is assumed that the two masses are located in the x_A axis thus $\mathbf{r}_1 = [x_1, 0, 0]^T$ and $\mathbf{r}_2 = [x_2, 0, 0]^T$. By the definition of center of mass the identity $x_1\mu_1 + x_2\mu_2 = 0$ must hold. Note that in the A frame the masses coordinates are constant thus gravity only depends on position coordinates as $\mathbf{a}_{\text{grav}} \equiv \mathbf{a}_{\text{grav}}(\mathbf{r})$. Consequently Eq. (1) denotes an autonomous system that does not depend on time t. Note that, unless indicated otherwise, vectors are expressed in the A frame by default.

2.3 Asteroid shape

Since this work concerns with safe orbits, a description of the asteroid surface has to be made. Specifically, we seek for a function that takes position coordinates as input and outputs if the evaluation point is interior or exterior to the shape. While an irregular polyhedron model [12] can be used for that purpose, it uses a binary variable to determine interior and exterior points (the normalized Laplacian) which is not differentiable. Alternatively, in this work the surface of the asteroid shape is approximated by an ellipsoid as

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} - 1 = 0 \tag{4}$$

The spacecraft is safe when the left-hand side after evaluating Eq. (4) is a positive number. Alternatively, the spacecraft collides when it crosses the zero of the ellipsoid function.

3 Optimal control for safe orbits

In this section, the continuous optimal control problem (OCP) is formulated and a penalty term for abandoning a safe orbital shell is designed.

3.1 Problem formulation

The problem of safely orbiting a central body is posed as the optimal control problem (5). In this work, the control time-horizon is considered fixed to be t_f. The number of impulses n and the instants

t_k of application are fixed a-priori and are not subject to optimization. Accordingly, the single decision variable is the impulse vector $\mathbf{v}_k$ at instants t_k.

$$
\begin{aligned}
\underset{\Delta\mathbf{v}_k}{\text{minimize}} \quad & J = \frac{1}{n}\sum_{k=0}^{n-1}\|\Delta\mathbf{v}_k\|_1 + \frac{\lambda}{t_f}\int_0^{t_f} L(\mathbf{r})\,dt \\
\text{subject to} \quad & \dot{\mathbf{r}} = \mathbf{v}, \quad t \in [0, t_f], \\
& \dot{\mathbf{v}} = -\boldsymbol{\omega}\times(\boldsymbol{\omega}\times\mathbf{r}) - 2\boldsymbol{\omega}\times\mathbf{v} - \mu_1\frac{\mathbf{r}-\mathbf{r}_1}{\|\mathbf{r}-\mathbf{r}_1\|^3} - \mu_2\frac{\mathbf{r}-\mathbf{r}_2}{\|\mathbf{r}-\mathbf{r}_2\|^3}, \\
& \mathbf{v}(t_k^+) = \mathbf{v}(t_k^-) + \Delta\mathbf{v}_k, \quad k = 0\ldots n-1, \\
& \mathbf{r}(0) = \mathbf{r}_0, \quad \mathbf{v}(0) = \mathbf{v}_0, \\
& \frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} \geq 1, \quad \|\mathbf{r}\| \leq r_{\max}, \quad \forall t \in [0, t_f], \\
& -\Delta\mathbf{v}_{\max} \leq \Delta\mathbf{v}_k \leq \Delta\mathbf{v}_{\max}
\end{aligned}
\tag{5}
$$

In the optimal control problem formulated in (5), the goal is to minimize the mean L1-norm of control impulses. There is an additional cost term $L(\mathbf{r})$ that penalizes being outside an orbital shell. The weight λ is a positive scalar that measures the relative importance given of the shell penalty with respect to fuel consumption. This penalty is further explained in the subsequent section. In terms of constraints, the solution trajectory must depart from the spacecraft initial state $(\mathbf{r}_0, \mathbf{v}_0)$ following the asteroid dynamics described by Eq. (1)-(3). Path constraints are considered via collision avoidance with the asteroid and by establishing an upper limit to the orbital radius to avoid escapes. To avoid collisions with the asteroid, the ellipsoid equation given by Eq. (4) is used as an inequality. In terms of escaping the asteroid gravity field, an admissible solution would be to keep the orbit eccentricity below the unity. However, that could still lead to large altitudes that may hinder the close proximity exploration of the asteroid. In that sense, it is enforced that the spacecraft trajectory remains below a maximum user-defined orbital radius $r_{\max}$. Lastly, the impulses are also constrained to a maximum amplitude threshold defined by $\Delta\mathbf{v}_{\max}$.

Note that the OCP defined by (5) is non-convex due to the asteroid collision avoidance inequality. Geometrically, it can be easily deduced that antipodal points outside the ellipsoid can be connected by a straight line that trespasses the forbidden region. Consequently, there may exist several local minima of problem (5). However, finding the global minimum is out of the scope of this work.

3.2 Orbital shell penalty

If $\lambda = 0$ is chosen in the objective function of problem (5), the single optimization goal is to minimize the required impulse amplitudes to satisfy path constraints. Therefore, it is very likely that the resulting trajectory touches the frontier of the path constraints being prone to trespassing when disturbances are incorporated. Since that behaviour is undesirable from the robustness point of view, the idea of this work is to foster the trajectory to be within an inner and outer orbital radius. This is what the integral cost modulated by the instantaneous penalty $L(\mathbf{r})$ represents in the OCP (5).

In this work, the design of the function $L(\mathbf{r})$ is made to keep the trajectory within a spherical shell with an inner radius r_{in} and an outer radius r_{out}. The L function has to provide a constant null value within the orbital shell and increase monotonically outside the shell region. In order for the L function to fulfill the previous requirements while being smooth and continuous, the following structure is proposed

$$L(r) = \kappa(h(\bar{r}+1) + h(\bar{r}-1) - \bar{r}) - L_0, \quad \bar{r} = \frac{2(r - r_{\text{in}})}{r_{\text{out}} - r_{\text{in}}} - 1 \tag{6}$$

where $r = \|\mathbf{r}\|$ is the orbital radius, $(r_{\text{in}}, r_{\text{out}})$ defines the inner and outer radius of the orbital shell. The variable κ is a user-defined parameter that controls the slope outside the shell and L_0 is an offset chosen such that makes $L((r_{\text{in}} + r_{\text{out}})/2) = 0$ thus $L(r) \geq 0 : \ \forall r \in \mathbb{R}$. The term h is the softplus function which is defined by

$$h(x) = \frac{1}{\beta} \log(1 + e^{\beta x}) \tag{7}$$

where β is an important tuning parameter that controls the plateau of the $L(r)$ function within the orbital shell region. A proper balance has to be ensured between plateauing L within the orbital shell and avoiding a sharp change of the function near the shell bounds. Figure 2 shows the function $L(r)$ curve for different values of β. This may help to choose an adequate value of β by taking into account the previous considerations.

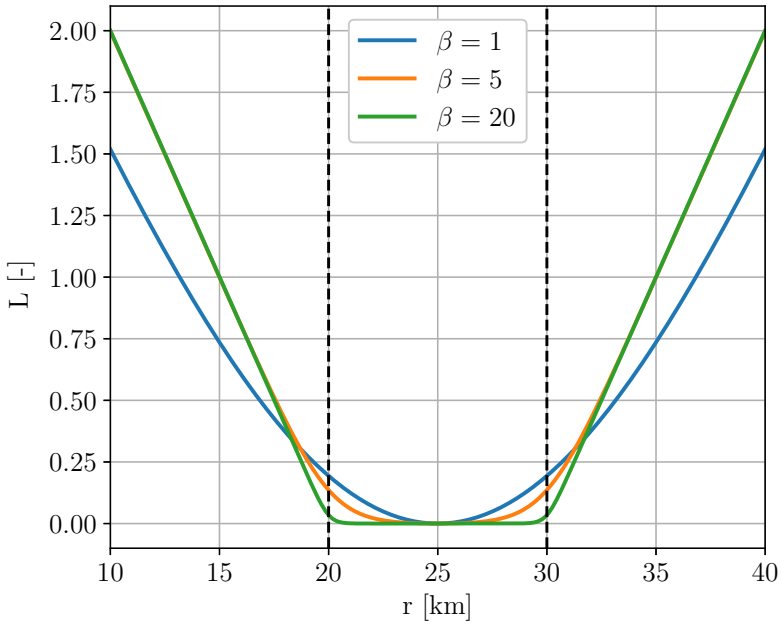

Fig. 2 Orbital shell penalty function for $(r_{\text{in}}, r_{\text{out}})$=(20,30) km and $\kappa = 1$.

4 Optimal control algorithms

In this section, two approaches for solving the optimal control problem (OCP) defined in (5) are presented. The first approach, direct transcription, relies on discretization to formulate the problem as a nonlinear program. The second approach, soft actor-critic (SAC), employs an exploration–exploitation strategy to learn a global control policy. The key design element of the latter method is the reward function, which must accurately capture the original optimal control objective in (5).

4.1 Direct Optimal Control

The direct transcription method for an OCP relies on discretizing the original continuous problem (5) into a static non-linear program which can be iteratively solved. There is a certain loss of accuracy, when compared to indirect methods, since cost, constraints and dynamics are only enforced at discrete points. However, the direct method is way more flexible since it allows to incorporate or remove features with minimal changes in the formulation.

To proceed with the discretization, let us define the spacecraft state at the jth time node t_j as $\mathbf{x}_j = [\mathbf{r}^T(t_j), \mathbf{v}^T(t_j)]^T$. The time node is $t_j = j\dfrac{t_f}{n_x}$. To treat the dynamics constraints in a static form, it is assumed that dynamics integration boils down to algebraic relations such that $\mathbf{x}_j \approx \mathbf{x}_{j-1} + \mathbf{f}_{\mathrm{RK}}(\mathbf{x}_{j-1})$ where $\mathbf{f}_{\mathrm{RK}}$ is an algebraic integration rule for Eq. (1)-(3) based on a Runge-Kutta scheme. By introducing the previous discretization into the original OCP (5) yields

$$
\begin{aligned}
\underset{\mathbf{x}_j,\, \Delta\mathbf{v}_k^+,\, \Delta\mathbf{v}_k^-}{\text{minimize}} \quad & J = \frac{1}{n}\sum_{k=0}^{n-1}\left(\|\Delta\mathbf{v}_k^+\|_1 + \|\Delta\mathbf{v}_k^-\|_1\right) + \frac{\lambda}{n_x}\sum_{j=1}^{n_x} L(\mathbf{r}_j) \\
\text{subject to} \quad & \mathbf{x}_j^- = \mathbf{x}_{j-1}^+ + \mathbf{f}_{\mathrm{RK}}(\mathbf{x}_{j-1}^+), \quad j = 1 \dots n_x, \\
& \mathbf{x}_j^+ = \mathbf{x}_j^- + \mathbf{B}\Delta\mathbf{v}_k, \qquad j = n, 2n \dots \frac{n_x}{n}n, \qquad k = 1 \dots n, \\
& \mathbf{x}(0) = \mathbf{x}_0, \\
& \frac{x_j^2}{a^2} + \frac{y_j^2}{b^2} + \frac{z_j^2}{c^2} \geq 1, \quad \|\mathbf{r}_j\| \leq r_{\max}, \quad j = 1 \dots n_x, \\
& \Delta\mathbf{v}_k = \Delta\mathbf{v}_k^+ - \Delta\mathbf{v}_k^-, \\
& \mathbf{0} \leq \Delta\mathbf{v}_k^+ \leq \Delta\mathbf{v}_{\max}, \\
& \mathbf{0} \leq \Delta\mathbf{v}_k^- \leq \Delta\mathbf{v}_{\max}
\end{aligned}
\tag{8}
$$

Note that the impulse $\Delta\mathbf{v}_k$ decision variable is augmented with its positive $\Delta\mathbf{v}_k^+$ and negative $\Delta\mathbf{v}_k^-$ components to avoid the L1-norm discontinuity when crossing from positive to negative values and vice versa. The control matrix is $\mathbf{B} = [\mathbf{0}_{3\times3}, \mathbf{I}]^T$. The number of evaluation states is n_x which is a multiple of the number of impulses n such that $n_x = in$, $i \in \mathbb{N}$. This may allow to enforce path constraints with a higher resolution. Due to the previous considerations the number of decision variables is $6n + 6n_x$. The number of inequality constraints is $2n_x + 6n$ accounting for path constraints and impulse amplitude bounds while equality constraints amount to $6n_x$ from dynamics enforcement. The non-linear program described by (8) can be iteratively solved, from an initial guess, by interior point algorithms which are implemented in several off-the-shelf numerical libraries.

4.2 Deep RL: Soft-Actor Critic

There exists several deep reinforcement learning (RL) algorithms compatible with different state-action types of variable. In the problem under consideration, both state (spacecraft position and velocity) and action (spacecraft delta-v) spaces are continuous and multidimensional. A deep RL algorithm that can handle this scenario is the soft actor-critic (SAC) [10] which is used in this work. Any RL algorithm requires a description of the environment as a Markov decision process (MDP). Translated to the problem under consideration (5), its MDP is described by the tuple $(\mathcal{X}, \Delta\mathcal{V}, \mathcal{P}, \mathcal{R})$ where $\mathcal{X}$ and $\Delta\mathcal{V}$ are the continuous state and action spaces. The term $\mathcal{P}$ represents the state transition probability $\mathcal{P} : \mathcal{X} \times \mathcal{X} \times \Delta\mathcal{V} \to [0, \infty)$ of the next state $\bar{\mathbf{x}}_{k+1} \in \mathcal{X}$ given the current state $\bar{\mathbf{x}}_k \in \mathcal{X}$ and action $\Delta\mathbf{v}_k \in \Delta\mathcal{V}$. The environment provides a bounded reward $\mathcal{R} : \mathcal{X} \times \Delta\mathcal{V} \to [r_{\min}, r_{\max}]$ on each state transition. Note the reward symbol r is the same as the orbital radius and should not be confused with it.

In this work, the MDP state is augmented as with the past cumulative delta-v expenditure as $\bar{\mathbf{x}}_k = [\mathbf{r}_k^T, \mathbf{v}_k^T, \sum_{i=0}^{k-1}\|\Delta\mathbf{v}_i\|_1]^T$. The previous choice allows to directly inform the RL agent about control consumption. The main goal in RL is to find an optimal policy function $\pi(\Delta\mathbf{v}_k|\bar{\mathbf{x}}_k)$ that maps the spacecraft state to an impulse vector at time t_k. Standard RL algorithms seek to find the policy that

maximizes the expected sum of rewards. However, SAC introduces the optimization of an additional maximum entropy objective for the policy as

$$\max_{\pi} J(\pi) = \sum_{k=0}^{n} \gamma^k \mathbb{E}_{(\bar{\mathbf{x}}_k, \Delta\mathbf{v}_k)\sim\rho_\pi}[r(\bar{\mathbf{x}}_{k+1}, \Delta\mathbf{v}_k) + \alpha\mathcal{H}(\pi(\cdot|\bar{\mathbf{x}}_k))] \tag{9}$$

where ρ_π is the time-weighted probability of visiting each state-action pair under a policy as it generates trajectories through the environment. While the environment natural dynamics is deterministic as per Eq. (1)-(4), the policy is stochastic. The term $\gamma \in (0, 1]$ is the discount factor that may give more relative importance to short-term rewards. Finally, $\mathcal{H}$ is the policy statistical entropy and α is the temperature parameter that weights the relative importance given to the policy entropy against the cumulative reward. In the seminal SAC paper [10], it is demonstrated that the policy entropy incentives exploration during training and derives a more robust policy against disturbances in evaluation mode. Note that deep RL deals with a maximization problem as per Eq. (9) while OCP defines a minimization problem (5).

To find a suitable policy that maximizes Eq. (9), knowledge of the action-value function $Q : \mathcal{X} \times \Delta\mathcal{V} \to \mathbb{R}$ is required. The function Q is denoted as

$$Q^\pi(\bar{\mathbf{x}}, \Delta\mathbf{v}) = \mathbb{E}\left[\sum_{k=0}^{n} \gamma^k \Big(r(\bar{\mathbf{x}}_{k+1}, \Delta\mathbf{v}_k) - \alpha \log \pi(\Delta\mathbf{v}_k|\bar{\mathbf{x}}_k)\Big) \;\Big|\; \bar{\mathbf{x}}_0 = \bar{\mathbf{x}},\ \Delta\mathbf{v}_0 = \Delta\mathbf{v}\right] \tag{10}$$

which represents the expected sum of rewards by following the policy $\pi(\Delta\mathbf{v}_k|\bar{\mathbf{x}}_k)$ after choosing the action $\Delta\mathbf{v}_0 \in \Delta\mathcal{V}$ from state $\bar{\mathbf{x}}_0 \in \mathcal{X}$. In SAC, the policy π is denoted as the actor because it defines an action to be taken at a certain state. The function Q^π is denoted as the critic because it quantifies the long-term outcome of the previous state-action pair. The addition of the policy entropy term, through the negative log-probability, makes Q^π to be denoted as a soft Q-function. If the soft Q-function is available, the optimal policy can be determined by using the Boltzmann soft-policy formula

$$\pi^*(\Delta\mathbf{v}|\bar{\mathbf{x}}) = \frac{\exp\left(\frac{1}{\alpha}Q^\pi(\bar{\mathbf{x}}, \Delta\mathbf{v})\right)}{\displaystyle\int_{\Delta\mathcal{V}} \exp\left(\frac{1}{\alpha}Q^\pi(\bar{\mathbf{x}}, \Delta\mathbf{v})\right) d\Delta\mathbf{v}} \tag{11}$$

which states that an optimal stochastic policy π^* will more likely choose actions with a better expected return in terms of the soft Q-function.

4.2.1 Learning the Q-function and policy

In practice, function approximators of the policy $\pi(\Delta\mathbf{v}_k|\bar{\mathbf{x}}_k)$ and soft-Q function are required. An usual choice is to describe both functions by using deep neural networks with parameters ϕ and θ respectively. Hence, let us denote the parameterized policy as $\pi_\phi(\Delta\mathbf{v}_k|\bar{\mathbf{x}}_k)$ and the tractable soft Q-function as $Q_\theta(\bar{\mathbf{x}}_k, \Delta\mathbf{v}_k)$. Typically, the policy is modeled as a multivariate Gaussian distribution with its mean and diagonal covariance computed by the deep neural network. The main difficulty is that the soft Q-function is a-priori unknown and has to be approximated by exploration. SAC is an off-policy learning algorithm that fills a replay buffer $\mathcal{D}$ composed of transition tuples $(\bar{\mathbf{x}}_k, \Delta\mathbf{v}_k, r_k, \bar{\mathbf{x}}_{k+1})$. The replay buffer, which may be non-empty initially, is augmented during training by storing experiences from different episodes.

The soft Q-function is subject to the following recursion widely known as the Bellman equation

$$Q_\theta(\bar{\mathbf{x}}_k, \Delta\mathbf{v}_k) \;=\; r(\bar{\mathbf{x}}_{k+1}, \Delta\mathbf{v}_k) + \gamma\, \mathbb{E}_{\bar{\mathbf{x}}_{k+1}=\bar{\mathbf{x}}_k^+ + \mathbf{f}_{\mathrm{RK}}(\mathbf{x}_k^+),\, \Delta\mathbf{v}_{k+1}\sim\pi_\phi(\cdot|\bar{\mathbf{x}}_{k+1})}\Big[Q_\theta(\bar{\mathbf{x}}_{k+1}, \Delta\mathbf{v}_{k+1}) - \alpha \log \pi_\phi(\Delta\mathbf{v}_{k+1}|\bar{\mathbf{x}}_{k+1})\Big]$$

which states that the soft Q-function of the current state-action pair is the immediate reward plus the discounted expected value of the next state-action pair adjusted by the entropy bonus. Initially, the soft Q-function approximation by the deep neural network Q_θ does not fulfill Bellman equation. To find the Q-function parameters θ, the temporal difference error defined by Bellman equation is minimized over a mini-batch $\mathcal{B} \subset \mathcal{D}$ of the replay buffer. In particular, the loss function for the soft Q network is

$$\mathcal{L}_Q(\theta) = \mathbb{E}_{(\bar{\mathbf{x}}_k, \Delta\mathbf{v}_k)\sim\mathcal{B}}\left[\frac{1}{2}\Big(Q_\theta(\bar{\mathbf{x}}_k, \Delta\mathbf{v}_k) - y_k\Big)^2\right] \tag{12}$$

where y_k is the Q-target defined by the recursive Bellman equation as

$$y_k = r(\bar{\mathbf{x}}_{k+1}, \Delta\mathbf{v}_k) + \gamma\, \mathbb{E}_{\Delta\mathbf{v}_{k+1}\sim\pi_\phi(\cdot|\bar{\mathbf{x}}_{k+1})}\Big[\bar{Q}_{\bar{\theta}}(\bar{\mathbf{x}}_{k+1}, \Delta\mathbf{v}_{k+1}) - \alpha \log \pi_\phi(\Delta\mathbf{v}_{k+1}|\bar{\mathbf{x}}_{k+1})\Big] \tag{13}$$

and $\bar{Q}_{\bar{\theta}}$ is a slowly-varying soft Q-function that circumvents the bootstrapping of Bellman equation. Note that the Q-targets are formed by using the Q-function which is the variable to be learned. The parameters of the slowly-varying Q-function $\bar{Q}_{\bar{\theta}}$ are slowly updated by $\bar{\theta} \leftarrow \tau\theta + (1-\tau)\bar{\theta}$.

The policy parameters are trained to exploit actions that drives the agent towards high values of the Q-function while keeping an adequate level of entropy. Accordingly, the policy loss combines the previous goals as

$$\mathcal{L}_\pi(\phi) = \mathbb{E}_{\bar{\mathbf{x}}_k\sim\mathcal{B},\, \Delta\mathbf{v}_k\sim\pi_\phi(\cdot|\bar{\mathbf{x}}_k)}\Big[\alpha\ \log \pi_\phi(\Delta\mathbf{v}_k|\bar{\mathbf{x}}_k) - Q_\theta(\bar{\mathbf{x}}_k, \Delta\mathbf{v}_k)\Big] \tag{14}$$

Note that the policy parameters π enter in the evaluation of the soft-Q function since actions are drawn according to the policy as $\Delta\mathbf{v}_k \sim \pi_\phi(\cdot|\bar{\mathbf{x}}_k)$. One should note that losses are minimized while RL objective function Eq. (9) is maximized, hence the discrepancy in the sign of some variables in Eq. (14).

Lastly, the temperature coefficient α is subject to discussion. In the seminal SAC paper [10], the temperature coefficient is fixed and is manually tune to achieve a desirable tradeoff between exploration and exploitation. In a subsequent version [13], the automatic adjustment of α to a desirable target entropy $\mathcal{H}_{\text{target}}$ was introduced. Following the auto-tuned temperature coefficient version, a loss function for α is defined by

$$\mathcal{L}_\mathcal{H}(\alpha) = \mathbb{E}_{\bar{\mathbf{x}}_k\sim\mathcal{B},\, \Delta\mathbf{v}_k\sim\pi_\phi(\cdot|\bar{\mathbf{x}}_k)}\Big[-\alpha\,\big(\log \pi_\phi(\Delta\mathbf{v}_k|\bar{\mathbf{x}}_k) + \mathcal{H}_{\text{target}}\big)\Big] \tag{15}$$

which will decrease the value of α if the policy entropy is higher than the target entropy and vice versa.

4.2.2 SAC training and evaluation

SAC is an off-line policy method which can learn from data generated by any policy. In that sense, the agent experiences MDP tuples $(\bar{\mathbf{x}}_k, \Delta\mathbf{v}_k, r_k, \bar{\mathbf{x}}_{k+1})$ which are stored in a replay buffer $\mathcal{D}$ over time. Simulations of the environment under consideration, Eq. (1)-(4), are sequentially done while learning the policy and Q-function. These simulations are named as episodes. The initial state of each episode is reset by randomly choosing a sample within an initial set $\bar{\mathbf{x}}_0 \in \mathcal{X}_0$. On each step of an episode, an action $\Delta\mathbf{v}_k$ is sampled from the policy at each state $\bar{\mathbf{x}}_k$, the state-action pair transitions to the next state $\bar{\mathbf{x}}_{k+1}$ and a reward r_k is collected from the environment. Afterwards, by using a mini-batch from the replay buffer, the Q-function, policy and temperature coefficient are updated following Eq. (12)-(15).

The implementation of the SAC is summarized in Algorithm 1. In practice, the SAC algorithm keeps track of two separate Q-function approximators Q_{θ_1} and Q_{θ_2} to mitigate the potential overestimation bias caused by errors. This is done by forming the y_k targets with the minimum value of the two Q-functions as can be observed in line 12. Line 17 states the Polyak averaging to update the slowly-varying target Q-functions for the next episode.

Algorithm 1 Soft Actor-Critic

Initialize Q-function networks $Q_{\theta_1}, Q_{\theta_2}$, policy network π_ϕ and temperature coefficient α
Initialize target Q-function networks $\bar{Q}_{\theta_1}, \bar{Q}_{\theta_2}$.
Initialize replay buffer $\mathcal{D}$.
for each episode **do**
    Reset initial state $\bar{\mathbf{x}}_0 \in \mathcal{X}_0$
    for each environment step **do**
        Sample action from state $\Delta\mathbf{v}_k \sim \pi_\phi(\cdot|\bar{\mathbf{x}}_k)$
        Apply action $\Delta\mathbf{v}_k$ and do the state transition $\bar{\mathbf{x}}_{k+1} = \bar{\mathbf{x}}_k + \mathbf{f}_{\mathrm{RK}}(\bar{\mathbf{x}}_k)$
        Collect reward $r_k = r(\bar{\mathbf{x}}_{k+1}, \Delta\mathbf{v}_k)$
        Store MDP tuple in replay buffer $\mathcal{D} \leftarrow \mathcal{D} \cup (\bar{\mathbf{x}}_k, \Delta\mathbf{v}_k, r_k, \bar{\mathbf{x}}_{k+1})$
        Sample mini-batch from replay buffer $\mathcal{B} \sim \mathcal{D}$
        Compute Q-targets y_k for mini-batch transitions

$$y_k = r_k + \gamma \min_{i=1,2} \bar{Q}_{\theta_i}(\bar{\mathbf{x}}_{k+1}, \Delta\mathbf{v}_{k+1}) - \alpha \log \pi_\phi(\Delta\mathbf{v}_{k+1}|\bar{\mathbf{x}}_{k+1}) \text{ with } (r_k, \bar{\mathbf{x}}_{k+1}) \sim \mathcal{B},\ \Delta\mathbf{v}_{k+1} \sim \pi_\phi(\cdot|\bar{\mathbf{x}}_{k+1})$$

        Update Q-function networks $\theta_i \leftarrow \theta_i - \lambda_Q \nabla_{\theta_i} \mathcal{L}_Q(\theta_i)$
        Update policy network $\phi \leftarrow \phi - \lambda_\pi \nabla_\phi \mathcal{L}_\pi(\phi)$
        Update temperature coefficient $\alpha \leftarrow \alpha - \dfrac{d\mathcal{L}_\mathcal{H}(\alpha)}{d\alpha}$
    end for
    Update target Q-function networks $\bar{\theta}_i \leftarrow \tau\theta_i + (1-\tau)\bar{\theta}_i$
end for

The SAC policy uses squashing and a reparameterization trick to map its network outputs to an action. The policy network outputs $(\boldsymbol{\mu}_\phi, \log\boldsymbol{\sigma}_\phi) \equiv \pi_\phi(\bar{\mathbf{x}}_k)$ where $\boldsymbol{\mu}_\phi$ is the Gaussian mean of the raw action and $\boldsymbol{\sigma}_\phi = \exp(\log\boldsymbol{\sigma}_\phi)$ its standard deviation. The raw action is sampled by $\mathbf{a}_{\mathrm{raw}} = \boldsymbol{\mu}_\phi + \boldsymbol{\sigma}_\phi \odot \boldsymbol{\epsilon}$ where $\boldsymbol{\epsilon} \sim N_3(\mathbf{0}, \mathbf{I})$ is a canonical multivariable Gaussian distribution. The final delta-v vector is recovered after squashing it as $\Delta\mathbf{v} = \Delta\mathbf{v}_{\max} \oplus \tanh(\mathbf{a}_{\mathrm{raw}})$. The hyperbolic tangent squashing enforces the maximum control amplitude. During evaluation mode, the policy is made deterministic as $\mathbf{a}_{\mathrm{raw}} = \boldsymbol{\mu}_\phi$ which mitigates catastrophic failures due to random sampling.

4.2.3 Reward function design

The design of an appropiate reward function is fundamental for the well-posed behavior of the RL agent. The reward function is chosen to be as aligned as possible to the original OCP (5)

$$r(\bar{\mathbf{x}}_{k+1}, \Delta\mathbf{v}_k) = -\underbrace{\frac{\|\Delta\mathbf{v}_k\|_1}{n}}_{\text{Fuel}} - \underbrace{\frac{\lambda L(\bar{\mathbf{x}}_{k+1})}{n}}_{\text{Orbital shell}} + \underbrace{g(\bar{\mathbf{x}}_{k+1})}_{\text{Collision/escape}} \tag{16}$$

where the reward $g(\bar{\mathbf{x}}_k)$ severely penalizes the violation of collision or escape constraints as

$$g(\bar{\mathbf{x}}_{k+1}) = \begin{cases} G & \text{if} \quad \dfrac{x_{k+1}^2}{a^2} + \dfrac{y_{k+1}^2}{b^2} + \dfrac{z_{k+1}^2}{c^2} \leq 1 \\ G & \text{if} \quad \sqrt{x_{k+1}^2 + y_{k+1}^2 + z_{k+1}^2} \geq r_{\max} \\ 0 & \text{if} \quad (\text{otherwise}) \end{cases} \tag{17}$$

where G has to be a negative scalar large enough with respect to consumption and orbital shell penalties to enforce stable orbits. It should be noted that collision or escape immediately triggers the episode termination.

5 Numerical Results

The performance of Section 4 control algorithms in is evaluated using a synthetic asteroid model inspired by Eros. The asteroid's shape resembles Eros's overall dimensions and is approximated as an ellipsoid with principal axes $a = 16$ km, $b = 8$ km, and $c = 5$ km. The gravity field, however, differs from that of the actual Eros, since no two-mass models are available. The total gravitational parameter of Eros is $\mu = 4.46276 \times 10^5$ m^3/s^2, which is arbitrarily divided between two point masses. The first mass represents 0.6μ and is located at $\mathbf{r}_1 = [5.33,\ 0,\ 0]^T$ km, while the second corresponds to 0.4μ and is located at $\mathbf{r}_2 = [-8,\ 0,\ 0]^T$ km. The asteroid rotates uniformly about its z_A-axis with a rotation period of 5.27 h.

5.1 Orbit stability analysis

To emphasize the relevance of the problem under consideration, an orbital stability analysis is performed. The initial conditions correspond to circular orbits, with the orbital elements varied according to a uniform random distribution within the ranges specified in Table 1. Note that the initial eccentricity is fixed at $e_0 = 0$ in all cases. Each simulation runs for 10 hours, and a total of 10,000 samples of the orbital elements are generated randomly.

Table 1 Ranges of initial orbital elements for stability analysis.

Orbital element	Range	Orbital element	Range
Semi-major axis (a_0)	$\mathcal{U}(18, 28)$ km	Eccentricity (e_0)	0
Inclination (i_0)	$\mathcal{U}(0°, 180°)$	RAAN (Ω_0)	$\mathcal{U}(0°, 360°)$
Argument of perigee (ω_0)	0°	True anomaly (ν_0)	$\mathcal{U}(0°, 360°)$

The results of the simulations are presented in Fig. 3. Three distinct behaviors can be identified: orbits that eventually collide with the asteroid; orbits for which the orbital radius becomes significantly larger than the initial value, satisfying $r > r_{\max} = 50$ km, which indicates a divergent behavior (escape in some cases); and orbits that neither collide nor exhibit excessive radial growth, which are considered stable for the duration of the simulation. According to this classification, among the 10,000 sampled orbits, 13.32% result in collisions, 11.00% diverge, and 75.68% remain stable. In Fig. 3, the axes represent the initial semi-major axis and orbital inclination of each sample, as these parameters exhibit the clearest pattern in terms of stability. Specifically, collisions and diverging trajectories cluster within the lower-left region of the plot, corresponding to low semi-major axes and direct orbits ($i_0 < 90°$), with equatorial orbits being the most unstable. A smaller region in the upper-left corner, corresponding to low semi-major axes and retrograde orbits, also leads to collisions. This stability analysis highlights the importance of accounting for unstable regions, as trajectories within these zones may result in a catastrophic loss of the mission.

5.2 Control algorithms setup

The controlled orbit simulations are conducted over a duration of 10 hours, consistent with the analysis of the natural dynamics presented in Section 5.1. An impulse vector is applied every 10 minutes, resulting in a total of $n = 60$ control actions. The maximum impulse magnitude is constrained by $\Delta\mathbf{v}_{\max} = [0.2,\ 0.2,\ 0.2]^T$ m/s. The orbital shell region is defined between $r_{\text{in}} = 22$ km and $r_{\text{out}} = 30$ km,

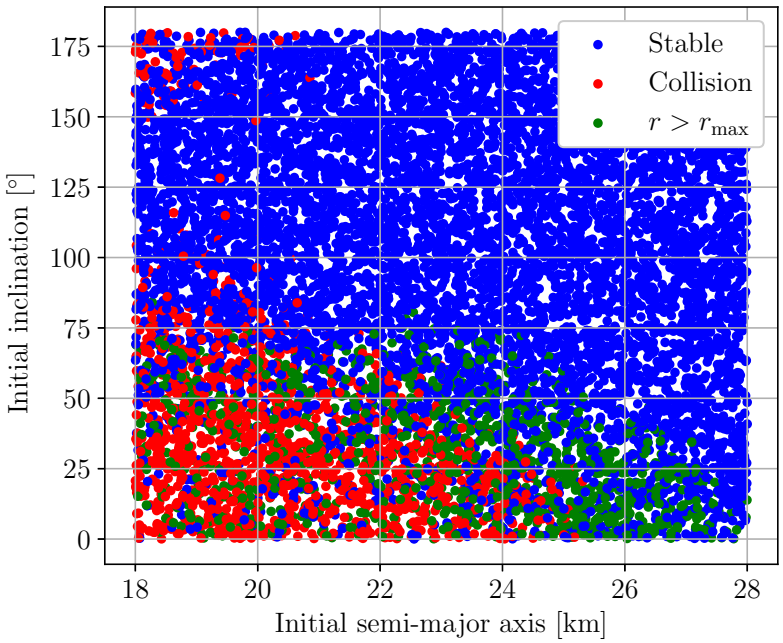

Fig. 3 Final outcome of initial circular orbits after 10 hours.

and a value of $\beta = 10$ was selected through ad-hoc experimentation to prevent gradient instability in the direct OCP method. The weighting factor for the orbital shell penalty is set to $\lambda = 0.1$, which was found to provide an adequate trade-off between path accuracy and delta-v expenditure.

Both the direct OCP and SAC approaches employ normalized variables in their internal computations. Specifically, dimensional positions are scaled by $R = 16$ km, corresponding to the asteroid's maximum elongation. Likewise, dimensional velocities and Δv values are scaled by $V = \sqrt{\mu/R} = 5.2813$ m/s, and time is normalized by $T = R/V = 50.49$ min. This normalization is consistently applied to the objective (or loss) functions, constraints, dynamics, and neural networks.

5.2.1 Direct Optimal Control

To implement the direct optimal control, CasADi [14] and IPOPT [15] libraries are used. CasADi writes the direct OCP problem of (8) by using symbolic variables and a 4th order Runge-Kutta integration scheme. While IPOPT iteratively solves the non-linear program from an initial guess. In this work, for systematicity, the initial guess corresponds to the natural trajectory resulting from $\mathbf{x}_0$ thus $\Delta\mathbf{v}_k^{[0]} = 0$.

5.2.2 Soft-Actor Critic

For the SAC implementation, the Stable-Baselines3 library [16] is employed. This framework enables seamless integration of a user-defined MDP environment with the standard SAC algorithm. The only modification introduced concerns the default Stable-Baselines3 behavior that prevents bootstrapping of truncated and terminal transitions when computing the Q-targets. In its original form, the algorithm disregards future rewards for both types of transitions. However, since truncation is artificially induced in this work, the default implementation was modified so that truncated transitions are treated equivalently to non-terminal ones during Q-target computation. This way the door is open for truncated transitions to receive future rewards. The SAC hyperparameters used in this study are summarized in Table 2. The majority of the hyperparameters follow the improved SAC publication [13] and the target entropy $\mathcal{H}_{\text{target}} = -\dim(\Delta\mathbf{v}) = -3$ is chosen to match the action dimension.

The terminal reward for collision and escape, as defined in Eq. (17), is set to $G = -5$. The SAC training process is limited to 300,000 interactions with the environment, corresponding to approximately

Table 2 Hyperparameters used in the Soft Actor-Critic (SAC) training.

Hyperparameter	Value	Hyperparameter	Value
Learning rates λ_Q, λ_π	3×10^{-4}	Discount factor (γ)	0.99
Replay buffer size	10^6	Mini-batch size	256
Target smoothing coeff. (τ)	0.005	Target entropy ($\mathcal{H}_{\text{target}}$)	-3
Hidden layers (Q_θ, π_ϕ)-networks	2 (ReLU)	Neurons per layer	256

5,000 truncated episodes. The actual number of training episodes exceeds this value, as some terminate prematurely due to escape or collision events. The evolution of the total reward per episode is illustrated in Fig. 4 for ten different random seeds. Convergence is observed after approximately 0.225 million steps, beyond which terminal episodes (with a total reward of ≈ -5) disappear entirely across all seeds. It should be noted that the policy is make deterministic during evaluation mode (after training is completed).

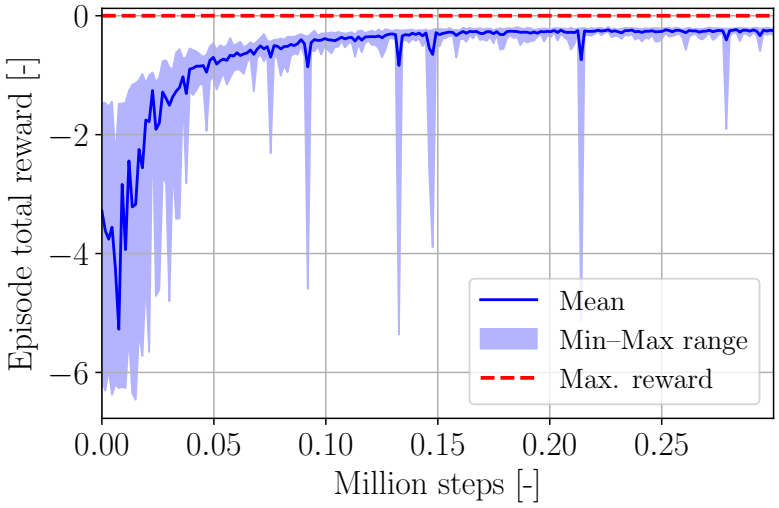

Fig. 4 Evolution of the episode total reward during SAC training for 10 different random seeds.

5.3 Controlled orbits

To evaluate the performance of the SAC and direct OCP algorithms, 500 cases from the stability analysis presented in Section 5.1 are sampled and used as a test set. From the 500 test samples, 386 are already stable, 59 collide with the asteroid and 55 escape. The same test samples are used for both the SAC and direct OCP simulations. Figure 5 illustrates the final outcomes for orbits controlled by the SAC algorithm, the direct OCP method, and the natural (uncontrolled) dynamics. The SAC algorithm achieves 100% orbital stability across all test cases. In contrast, the direct OCP method attains approximately 94% stability, failing to prevent collisions in about 6% of the samples. In these cases, the nonlinear programming solver was unable to identify a feasible control solution. Further investigation is warranted, as alternative initial trajectory guesses may enable the solver to converge to a valid solution. The superior performance of SAC in achieving full stability is likely attributable to its exploration–exploitation mechanism, which enhances robustness in safety-related outcomes. Nevertheless, it is worth noting that the direct OCP approach successfully stabilizes roughly 50% of the naturally colliding cases and 100% of the natural escape trajectories.

Figure 6 compares the natural, SAC, and direct OCP trajectories for three representative test cases corresponding to naturally stable, colliding, and escaping orbits. Interestingly, in the naturally stable case, the direct OCP trajectory nearly coincides with the natural one, suggesting negligible fuel consumption. In contrast, for the collision and escape cases, both the direct OCP and SAC controllers successfully prevent

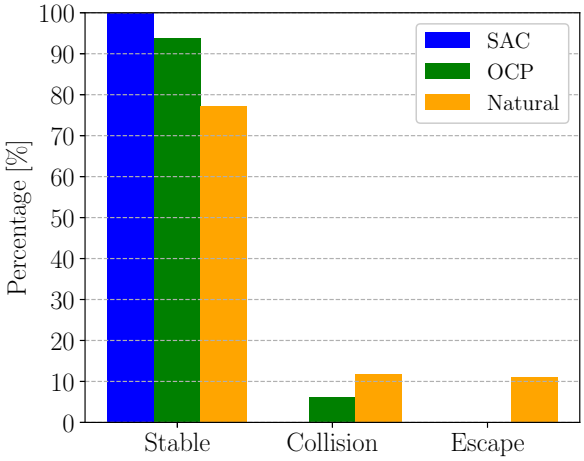

Fig. 5 Percentage of stable, collision and escape trajectories using soft-actor critic and direct optimal control for 500 test simulations.

the undesired outcomes observed in the natural dynamics. The Δv usage in the radial–tangential–normal components is shown in Fig. 7 for the same three cases. It can be observed that SAC consistently applies higher control effort across all scenarios, including the naturally stable trajectory. As further evidence across the complete test set will show, the direct OCP algorithm proves to be more efficient in terms of Δv expenditure than SAC. This behavior warrants additional investigation, as it may result from the stochasticity of the SAC policy introduced by the target-entropy term. Moreover, the inclusion of strong penalty terms in the reward function to enforce collision and escape avoidance could also compromise the accuracy of fuel consumption minimization.

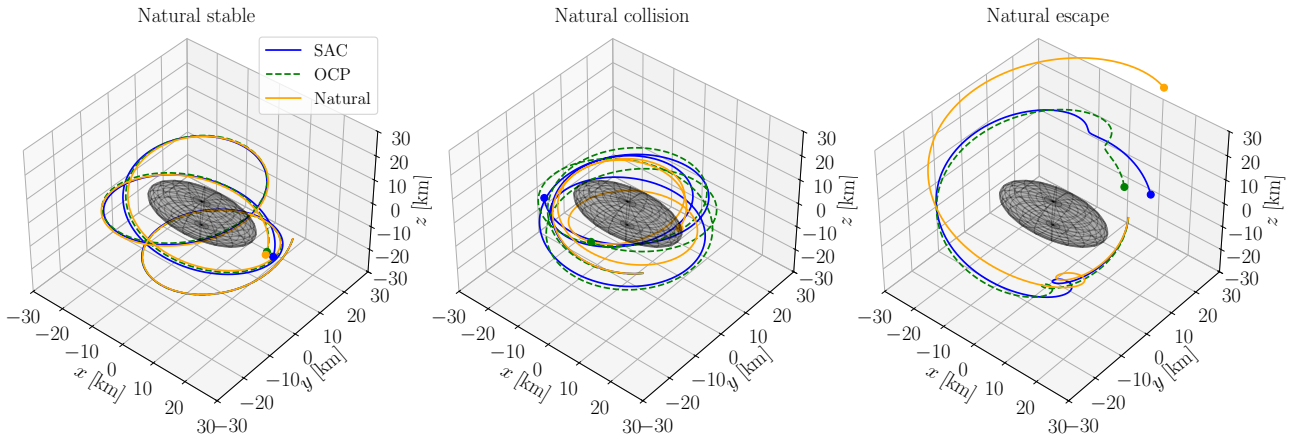

Fig. 6 Controlled orbits using soft-actor critic and direct optimal control.

To provide a broader overview of the test set behavior, the evolution of the orbital radius and the cumulative Δv expenditure are presented in Figs. 8–9. For visualization purposes, the analysis is limited to 50 representative test cases, corresponding to 10% of the total dataset. The orbital radius evolution shown in Fig. 8 indicates that the controlled trajectories, for both SAC and direct OCP, remain confined within the orbital shell between 22 and 30 km. Since the initial orbital radius of the test samples lie in the range of 18–28 km, the plots reveal that lower-altitude trajectories are raised into the desired shell region. More importantly, for both SAC and direct OCP, the orbits remain bounded within the shell without drifting toward the asteroid's center. This observation confirms that the penalty function defined in Eq. (6) is effective in maintaining orbital safety and has been properly tuned to avoid excessive corrective maneuvers. Figure 9 depicts the cumulative Δv consumption for both control approaches.

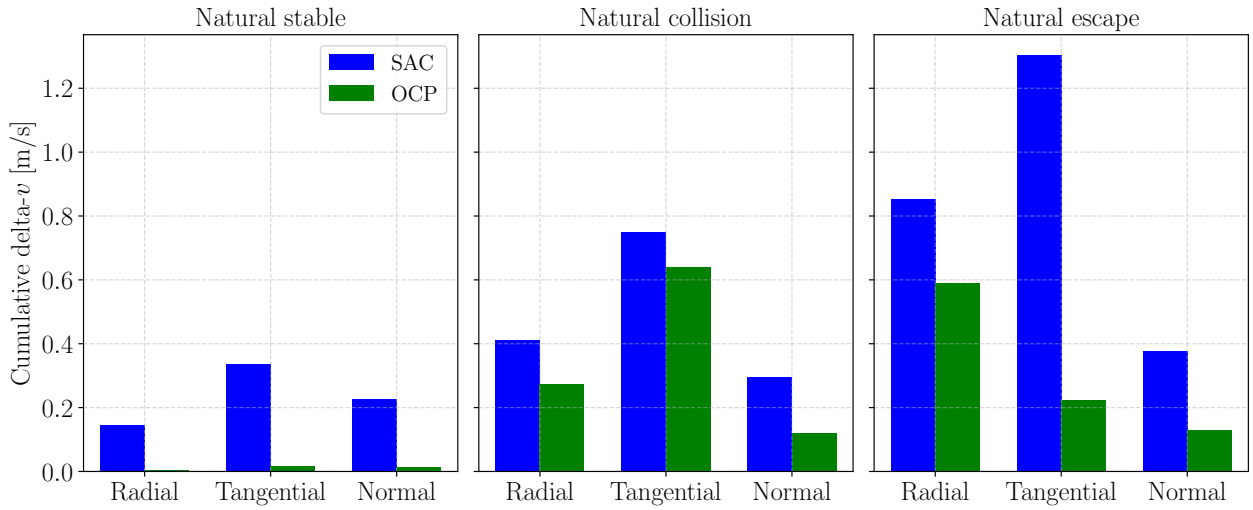

Fig. 7 Cumulative impulse in radial, tangential and normal components for controlled orbits using soft-actor critic and direct optimal control.

Overall, the SAC algorithm exhibits a total Δv expenditure in the range of 0.5–1 m/s, whereas the direct OCP maintains this value below approximately 0.25 m/s for most test cases. This trend is consistent with the results previously discussed in Fig. 7. It is also noteworthy that the direct OCP algorithm concentrates most of its control effort at the beginning of the simulation, while SAC displays a gradual, monotonic increase in Δv. This behavior further supports the hypothesis that the higher control effort observed in SAC may be linked to the stochasticity introduced by its policy entropy term.

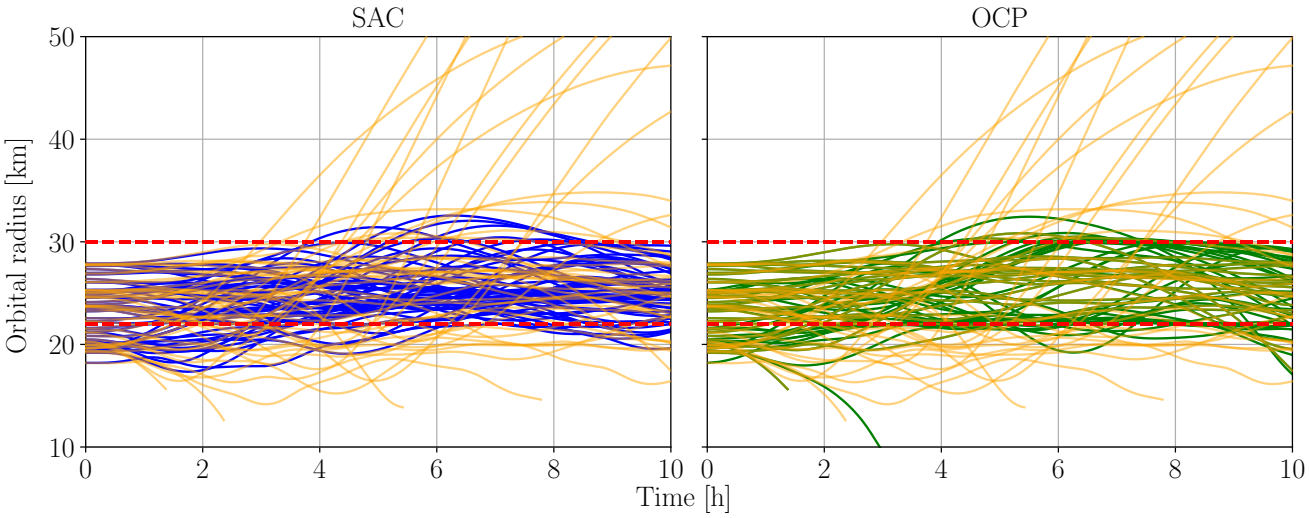

Fig. 8 Orbital radius evolution using soft-actor critic and direct optimal control for 50 test simulations.

Lastly, Table 3 reports the measured computational wall times on an Apple M2 Max CPU. Training the SAC algorithm requires approximately 40–45 minutes of wall-clock time, whereas direct OCP does not involve this training step. In a realistic deployment scenario, SAC training would typically be performed on Earth, with the resulting policy network uplinked to the spacecraft. Note that the presented results assume a perfect correlation between reality and training model. Accordingly, assessing the SAC robustness to cope with the reality gap is deemed as future work. The evaluation column shows the wall-clock time required to compute a control sequence on-the-fly for both SAC and direct OCP. For SAC, this time includes the n evaluations of the policy network, while for direct OCP it corresponds

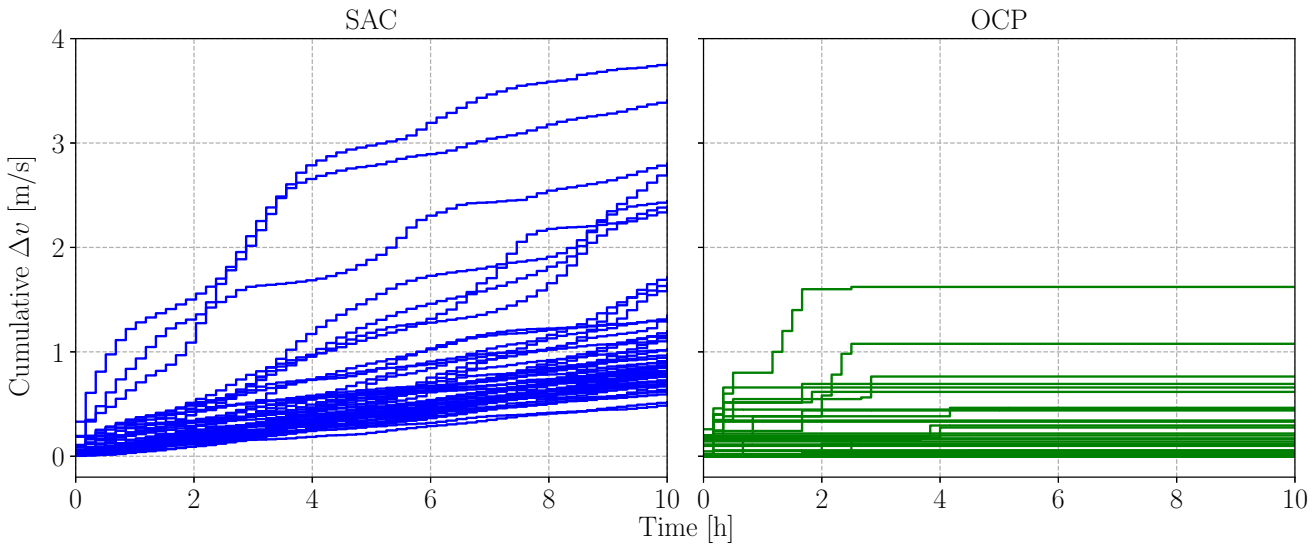

Fig. 9 Cumulative delta-v evolution using soft-actor critic and direct optimal control for 50 test simulations.

to the iterative solution of the non-linear program. Evaluating the policy network is very fast, since its architecture is just two hidden layers with 256 neurons, taking approximately 7–14 milliseconds. In contrast, solving the direct OCP non-linear program requires on average 0.5 minutes, with a worst-case scenario of 6.67 minutes. Although the software and hardware used in this study differ from those in an operational setting, these measurements provide insight into the relative computational burden of each algorithm.

Table 3 Computational wall-clock times for SAC and direct OCP control evaluation and training

	Evaluation		Training	
Wall-clock time	**Mean [s]**	**Maximum [s]**	**Mean [min]**	**Maximum [min]**
Soft-actor critic	$7.954{\cdot}10^{-3}$	$13.48{\cdot}10^{-3}$	42.67	44.17
Direct OCP	32.75	399.7	-	-

6 Conclusions

This manuscript addresses the problem of safely orbiting in highly perturbed environments, such as asteroid gravity fields. It formulates an optimal control problem for safe orbiting around asteroids and presents two solution approaches: one based on deep reinforcement learning and the other on direct transcription of the control problem. A novel aspect of this work is the inclusion of a penalty term that encourages the spacecraft to remain within a defined orbital shell. This avoids the ill-posed generation of controlled trajectories near collision or escape constraints while also preventing frequent delta-v corrections. The direct transcription method transforms the optimal control problem into a non-linear program that can be solved iteratively from an initial guess. Deep reinforcement learning, specifically using the soft actor-critic algorithm, approximates the state-value function and policy with deep neural networks through an exploration-exploitation training scheme. The direct transcription method is more fuel-efficient than SAC but fails to find feasible solutions for approximately 50% of the collision test cases. In contrast, SAC consumes more fuel due to its stochastic policy but successfully achieves stable controlled orbits across the entire test set. In terms of computational burden, SAC requires a substantial training overhead; however, once trained, evaluating the policy network is extremely fast compared to solving the direct OCP non-linear program on the fly.

Future work could focus on mitigating the relatively high fuel consumption of the soft actor-critic algorithm compared to direct optimal control. For this purpose, constrained SAC formulations, such as safety filters that verify whether a proposed policy action satisfies the constraints and provide a safe alternative if it does not, could be employed to avoid the soft-constraint approach, which may reduce fuel efficiency. In this context, the developed direct OCP algorithm could serve as a safety controller, although this would require training the policy network within the safe region from the very beginning. Another potential avenue is to train policy neural networks directly on solutions generated by the direct OCP, which may improve fuel efficiency while maintaining feasibility. The infeasibilities observed in the direct OCP approach also merit further investigation. Addressing this issue could involve exploring a broader range of initial guesses to guide the local iterative solver toward more feasible regions. Finally, the simulation environment could be extended to a higher-fidelity model that incorporates more realistic gravity fields, body shapes, and uncertainties from dynamics and control.

Acknowledgments

The authors gratefully acknowledge financial support from the Spanish Ministry of Science and Innovation under grant PID2023-147623OB-I00 and from the INDRA-University of Seville Space Surveillance Chair.

Declaration of Use of Artificial Intelligence

Generative AI has been used in the elaboration of this work to assist with programming tasks. Specifically, the authors have used ChatGPT-5 to assist with preliminary code structure, the parallelization of the simulations and plots customization.

References

[1] D. J. Scheeres. Orbit mechanics about asteroids and comets. *Journal of Guidance, Control, and Dynamics*, 35(3):987–997, 2012. doi: 10.2514/1.57247.

[2] Mark E. Holdridge. Near shoemaker spacecraft mission operations. *Johns Hopkins APL Technical Digest*, 23(1):58–67, January 2002. doi: 10.14339/JHATD.23.1.58.

[3] Daniel R. Wibben, Andrew Levine, James V. McAdams, Peter G. Antreasian, Samantha Rieger, and Kenneth M. Getzandanner. Osiris-rex orbit trim strategy. In *AIAA Scitech Forum*, 2021. NTRS Document ID: 20210025302; External Source DOI: 10.2514/6.2022-2471. `https://ntrs.nasa.gov/citations/20210025302`.

[4] Yang Yu and Hexi Baoyin. Generating families of 3d periodic orbits about asteroids. *Monthly Notices of the Royal Astronomical Society*, 427(1):872–881, 2012. doi: 10.1111/j.1365-2966.2012.21963.x.

[5] Hongwei Yang, Xiangyuan Zeng, and Hexi Baoyin. Feasible region and stability analysis for hovering around elongated asteroids with low thrust. *Research in Astronomy and Astrophysics*, 15(9):1223–1232, 2015. doi: 10.1088/1674-4527/15/9/004.

[6] Julio C. Sanchez, Rafael Vazquez, James D. Biggs, and Franco Bernelli-Zazzera. Orbit-attitude Predictive Control in the Vicinity of Asteroids with In Situ Gravity Estimation. *Journal of Guidance, Control, and Dynamics*, 45(3):262–279, 2022. doi: 10.2514/1.G005572.

[7] Rodolfo Batista Negri and Antonio F. B. A. Prado. Autonomous and robust orbit-keeping for small-body missions. *Journal of Guidance, Control, and Dynamics*, 2022. doi: 10.2514/1.G005863.

[8] J. Martin and H. Schaub. Reinforcement learning and orbit-discovery enhanced by small-body physics-informed neural network gravity models. In *AIAA Scitech 2022 Forum*, page 2272, 2022. doi: 10.2514/6.2022-2272, https://doi.org/10.2514/6.2022-2272.

[9] Hongyi Xie and Franco Bernelli-Zazzera. Event-triggered orbital and attitude station-keeping control for near-asteroid spacecraft. *Acta Astronautica*, pages 918–928, 2025. doi: 10.1016/j.actaastro.2025.01.053.

[10] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pages 1861–1870, 2018.

[11] T. Prieto-Llanos and M. A. Gómez-Tierno. Stationkeeping at libration points of natural elongated bodies. *Journal of Guidance, Control, and Dynamics*, 17(4):787–794, 1994. doi: 10.2514/3.21268.

[12] R. A. Werner and D. J. Scheeres. Exterior gravitation of a polyhedron derived and compared with harmonic and mascon gravitation representations of asteroid 4769 castalia. *Celestial Mechanics and Dynamical Astronomy*, 65(3):313–344, 1996. doi: 10.1007/BF00053511.

[13] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018.

[14] Joel A E Andersson, Joris Gillis, Greg Horn, James B Rawlings, and Moritz Diehl. CasADi – A software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation*, 2018.

[15] Andreas Wächter and Lorenz T. Biegler. On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1):25–57, 2006. doi: 10.1007/s10107-004-0559-y.

[16] Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22(268):1–8, 2021.