

1 Supplementary Material

2 1.1 OrbitZoo vs. Herrera: Environment Comparison

3 To compare OrbitZoo’s capabilities with those of existing environments that offer publicly available
4 code, we extend our analysis by implementing the mission proposed by Herrera in [1].

5 1.1.1 Definition

6 Herrera developed a reinforcement learning environment for station-keeping in Low Earth Orbit
7 (LEO). While Keplerian orbits describe a well-defined elliptical path around a central body, real-
8 world satellites are influenced by perturbative forces beyond gravity, causing them to drift from their
9 nominal orbits over time. Station-keeping missions aim to apply strategic control – through attitude
10 adjustments or thrust maneuvers – to maximize the duration a satellite stays within its designated
11 orbit. This task becomes especially challenging and critical in LEO due to the significant impact of
12 atmospheric drag.

13 1.1.2 Environment Setup

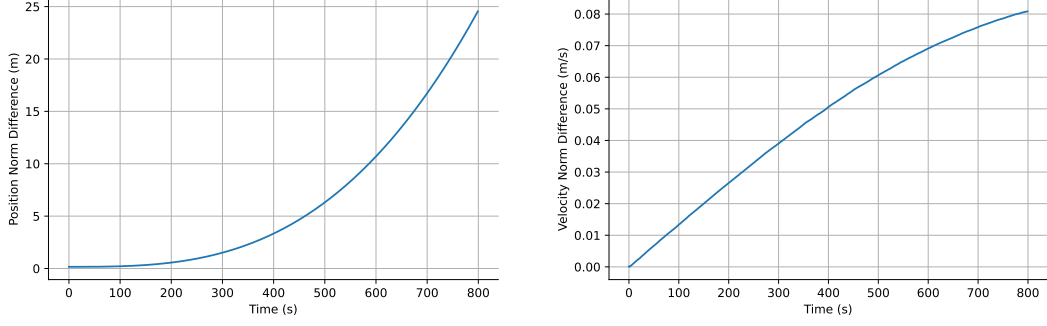
14 **Objective and Environment Characteristics.** In this mission, Herrera employed PPO to train
15 a satellite to control its thrust for station-keeping. The environment’s dynamics were manually
16 implemented using a 4th-order Yoshida integrator, which computes the satellite’s next position and
17 velocity based on gravitational and drag forces. The satellite began on a circular orbit at an altitude of
18 550 km above Earth’s surface, with the objective of maintaining that altitude for as long as possible
19 without deviating by more than 1 meter.

20 Herrera also clearly defined the episode termination conditions: (1) each episode has a maximum
21 duration of 800 steps, with each step representing 1 second; (2) if the satellite does not perform any
22 maneuvers, it exceeds the threshold after approximately 200 steps due to natural drift; and (3) the
23 satellite is provided with enough fuel to apply maximum thrust for up to 125 steps. For a mission
24 requiring this level of precision, the differences in dynamic modeling are critical. As stated by
25 Herrera: *"The 4th Order Yoshida integrator is used as the default integrator as there is a balance*
26 *of performance and error. RK4, while generally having less error, is not a semantic integrator and*
27 *requires the calculation of the acceleration 4 times while the Yoshida integrator only requires 3."*
28 Given the difference in integration methods – Herrera employing the faster but less precise Yoshida
29 integrator, and OrbitZoo using the more accurate Dormand-Prince integrator (a RK4 integrator
30 with adaptive time steps) – we conduct a direct comparison between the values produced by each
31 environment over the course of a full episode, starting from the same initial position and velocity. As
32 shown in Figure 1, the difference in position after approximately 13 minutes is around 25 meters,
33 while the difference in velocity is around 0.08 meters per second. With this comparison, we note that
34 the one meter threshold initially defined for episode termination is extremely strict under realistic
35 dynamic conditions. The code needed to run this comparison is provided in the supplementary
36 material.

37 Following Herrera’s setup, we designed a similar mission in which the agent has an initial mass of
38 100 kg, including 75 kg of fuel. The satellite is modeled as a perfect sphere with a radius of 16.8 meters,
39 a drag coefficient of 2.123 and is equipped with a thruster characterized by a specific impulse of
40 $I_{sp} = 0.0067$ s. While this configuration is not physically realistic, it was chosen to maintain
41 consistency with Herrera’s termination conditions (2) and (3). Notably, in OrbitZoo, atmospheric
42 drag is not solely determined by the satellite’s current state (as in Herrera’s implementation) but also
43 incorporates historical data, allowing drag to vary dynamically based on the current moment.

44 **Action Space.** To simplify the problem, Herrera limits the action space to the orbital plane,
45 and considers a polar thrust representation (T, θ) . However, the action the agent performs is
46 not directly the thrust being applied, but the variation applied to the current thrust, creating a
47 smoother and more realistic maneuver: $a_t = (\Delta T, \Delta \theta)$. While the maximum thrust is char-
48 acterized by $(T_{\max}, \theta_{\max}) = (0.04, 2\pi)$, the action space is limited to a fraction of that change:
49 $(\Delta T_{\max}, \Delta \theta_{\max}) = (T_{\max}/50, \theta_{\max}/6)$.

50 **Observation Space.** Similarly to Herrera, we use Cartesian coordinates to represent the current
51 position $r \in \mathbb{R}^2$ and velocity $\dot{r} \in \mathbb{R}^2$ of the satellite. Additionally, we also define the distance to the



(a) Euclidean distance between Herrera's and OrbitZoo's propagated position, per second.

(b) Euclidean distance between Herrera's and OrbitZoo's propagated velocity, per second.

Figure 1: Comparison between Herrera's and OrbitZoo's natural propagation: position and velocity differences over time.

Table 1: OrbitZoo vs. Herrera: Training hyperparameters.

Parameter	Value
Actor learning rate	0.0001
Critic learning rate	0.001
GAE λ	0.95
Epochs	5
Discount factor (γ)	0.99
Clip (ϵ)	0.03
Initial Standard Deviation	0.5
Standard Deviation Decay Rate	10000 steps
Standard Deviation Decay Amount	0.05
Minimum Standard Deviation	0.05
Experiences for Training	800
Batch Size	64

nominal altitude, $r_{\text{target}} = ||r|| - r_{\text{nominal}}$ and nominal velocity, $\dot{r}_{\text{target}} = ||\dot{r}|| - \dot{r}_{\text{nominal}}$, where

$$r_{\text{nominal}} = 550 \text{ km} \quad \dot{r}_{\text{nominal}} = \sqrt{\frac{\mu_E}{r_{\text{nominal}}}},$$

with μ_E representing the gravitational parameter of Earth. The observation $o_t \in \mathbb{R}^8$ consists of:

$o_t = (r/r_{\text{nominal}}, \dot{r}/\dot{r}_{\text{nominal}}, r_{\text{target}}, \dot{r}_{\text{target}}, \theta, T)$.

Reward Function. In order to encourage the agent to both not run out of fuel and stay within the orbital termination threshold, the reward function is similar to Herrera's:

$$r_t = \begin{cases} 0 & \text{if } r_{\text{target}} > 1\text{m} \vee f = 0 \\ \frac{t}{800} + 0.5 & \text{otherwise} \end{cases}, \quad (1)$$

where f is the current available fuel in kg, and t is the current step within the episode.

1.1.3 Results

Similarly to Herrera, we employed PPO to trained the agent, with the algorithm hyperparameters being shown in Table 1. If the agent can consistently exceed the baseline of 200 steps – representing the duration a satellite remains within tolerance without maneuvers – it demonstrates successful station-keeping. Figure 2 illustrates that, despite the narrow 1-meter tolerance from the nominal orbit (together with the more realistic perturbations), the agent effectively learned to maintain station-keeping by surpassing those 200 steps consistently.

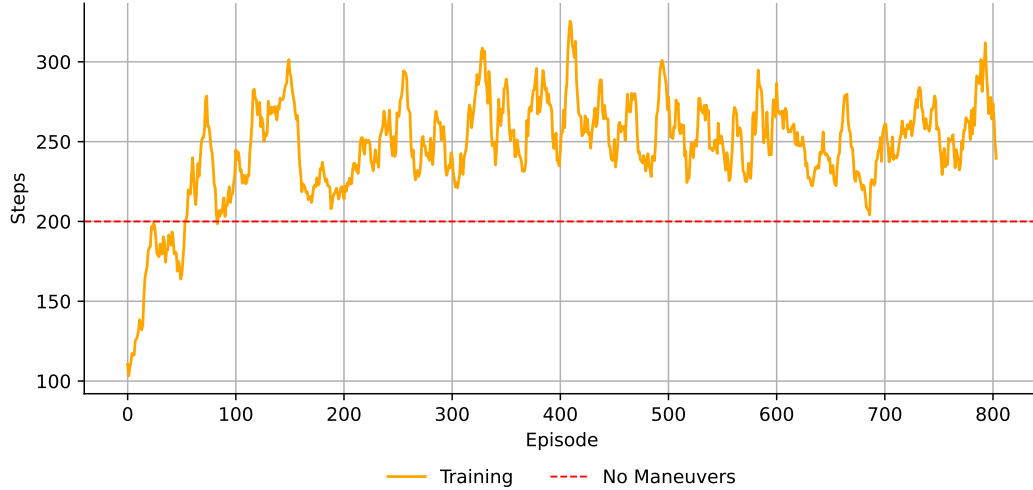


Figure 2: Steps per episode, averaged by a sliding window of 10 episodes. If no maneuvers are executed, the satellite exits the allowed tolerance range by step 200.

65 1.1.4 Challenges and Future Improvements

66 The agent did not achieve the same level of performance as in Herrera’s environment, primarily due to
 67 differences in the dynamics, as demonstrated earlier. Additionally, the PPO implementations differed
 68 – one using TensorFlow and the other PyTorch – with variations in hyperparameters.

69 This attempt to replicate Herrera’s environment suggests that OrbitZoo could benefit from using
 70 faster propagators. While these may be less precise, they are suitable for RL missions where extreme
 71 precision is not critical. As Herrera points out, continuous but deterministic algorithms (such as
 72 DDPG and TD3) may be better suited for this mission due to the precision required in controlling
 73 thrust maneuvers – something that is challenging to achieve with PPO.

74 **References**

- 75 [1] Armando Herrera III. Reinforcement learning environment for orbital station-keeping. Master's
76 thesis, The University of Texas Rio Grande Valley, 2020. ScholarWorks @ UTRGV.