Demonstration of Space Robot Teleoperation over a Lossy and Delayed Network using ATMOS ⋆

Inkyu Jang [1,*,†] **Gregorio Marchesini** [2,*] **Nicola De Carli** [2]
Byeongjun Kim [1] **Sunwoo Hwang** [1] **Dabin Kim** [1] **Elias Krantz** [2]
Youngkyoung Kong [1] **Frank J. Jiang** [3] **Annika Wong** [3]
Pedro Roque [4] **Prasetyo W. L. Sanjaya** [2] **Nicola Bastianello** [2]
Mani H. Dhullipalla [2] **Karl H. Johansson** [2] **Hyungbo Shim** [1]
Dimos V. Dimarogonas [2] **H. Jin Kim** [1]

[1] *Seoul National University, Seoul, Korea.*
[2] *KTH Royal Institute of Technology, Stockholm, Sweden.*
[3] *FleetMQ, Stockholm, Sweden.*
[4] *California Institute of Technology, Pasadena, CA, USA.*

Abstract: We present a demonstration showcasing the Autonomy Testbed for Multi-purpose Orbiting Systems (ATMOS), a planar spacecraft-analog robot designed for hardware-in-the-loop evaluation of guidance and control strategies in microgravity-like conditions. Using ATMOS as the physical test platform, we investigate the design, analysis, and performance evaluation of control architectures for remotely operated spacecraft under round-trip communication delays. In this work, we develop and experimentally validate a control strategy that combines state prediction and trajectory tracking control to perform a docking maneuver, accounting for time-varying random communication latency between ground operators and the ATMOS system. The demonstration includes a long-distance remote control experiment between Seoul and Stockholm, introducing realistic intercontinental delays and variability. The results highlight the capability of ATMOS to support rapid, reliable, and cost-effective testing of spacecraft teleoperation concepts, establishing a first step toward robust validation of on-orbit operations in microgravity-like environments.

Keywords: Aerial and Space Robotics, Control over Networks, Control under Communication Constraints, Remote Control, Aerospace Mission Control and Operations

1. INTRODUCTION

Recent technological advancements have fostered growing interest in the development of space missions for both terrestrial applications (e.g., climate monitoring, weather forecasting, gravitometry) and extraterrestrial exploration. As the demand for scientific throughput increases, remote-control architectures involving human-in-the-loop operators (Sheridan (1993); Chen and Liu (2021)) as well as networked multi-agent solutions (Liu and Kumar (2012); Zhang et al. (2018); Wang et al. (2019); Zhang et al. (2024)) have gained significant attention as reliable and cost-effective approaches to meet these mission requirements. These architectures depend on communication networks that may introduce delays, jitter, and packet loss, all of which can significantly degrade closed-loop performance.

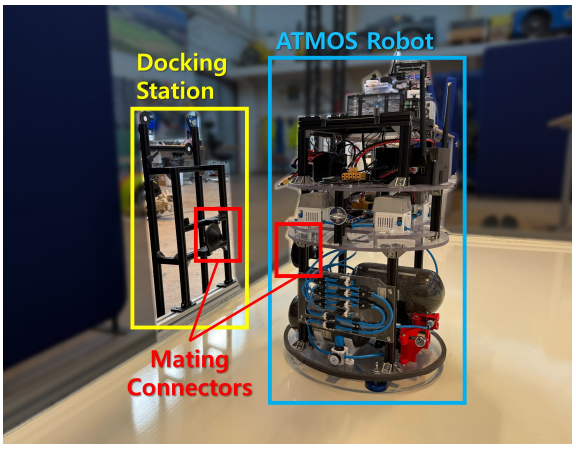


Fig. 1. The robot hardware (ATMOS, Roque et al. (2025)) considered in this demonstration.

The reliability and predictability of these communication channels are therefore essential for safety-critical spacecraft applications (Torgerson et al. (2024)). Designing

* Inkyu Jang and Gregorio Marchesini contributed equally to this work.

† Corresponding author. (e-mail: `janginkyu.larr@gmail.com`)

⋆ This work was supported in part by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (RS-2024-00436984), and in part by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg (KAW) Foundation, the Swedish Research Council (VR), and Digital Futures.

Fig. 2. An illustration of the physical distance of 7450 km between the two labs in Stockholm, Sweden (the ATMOS robot) and Seoul, Korea (the controller).

control strategies that remain effective in the presence of such imperfections requires experimental platforms that can reproduce realistic communication conditions while enabling safe and repeatable testing on the ground.

In this demonstration paper, we showcase the Autonomy Testbed for Multi-purpose Orbiting Systems (ATMOS, Roque et al. (2025), Fig. 1) as a robotic platform for the rapid and cost-effective testing of delay-resilient spacecraft control strategies. ATMOS is a planar free-flying space robot that enables hardware-in-the-loop experiments for guidance and control algorithms under realistic conditions.

We develop a delay-aware Control Lyapunov Function (CLF) based controller (Freeman and Primbs (1996)) capable of tracking both fixed and time-varying reference trajectories despite stochastic, time-varying communication delays. The main contribution of this work is that we present a transcontinental closed-loop control experiment between the space robotics laboratory in Kungliga Tekniska Högskolan (KTH) (Stockholm, Sweden) and Seoul National University (SNU) (Seoul, Korea). In this setup, control inputs are computed in Seoul and transmitted to the ATMOS robot in Stockholm, while state measurements are streamed back to Seoul. The communications over this 7450 km distance as shown in Fig. 2 are made through the FleetMQ [1] middleware. This introduces two-way latency profiles which affect the controller performance. Under the assumption that this latency comes from a slowly-varying distribution, we perform state prediction against delays based on previous observations of delay, and then apply the control in the direction to which the CLF value decays to achieve stability.

This demonstration highlights the capability of ATMOS to support the design and evaluation of delay-robust control architectures, providing an effective framework for validating teleoperation and autonomy concepts prior to on-orbit deployment.

2. SYSTEM MODEL

ATMOS (Roque et al., 2025) is a three degrees of freedom robotic platform moving quasi-frictionless on a resin plate to simulate motion in micro-gravity environments. The robot configuration is described by $\boldsymbol{q} = (\boldsymbol{p}, \psi) \in \mathbb{R}^2 \times (-\pi, \pi]$, where $\boldsymbol{p} \in \mathbb{R}^2$ denotes the robot position and $\psi \in (-\pi, \pi]$ represents the yaw angle of the robot. We indicate by $\dot{\boldsymbol{q}} = \begin{bmatrix}\boldsymbol{v}^\top & \omega_z\end{bmatrix}^\top \in \mathbb{R}^3$ the robot velocity twist, where $\boldsymbol{v} \in \mathbb{R}^2$ is the world-frame linear velocity and $\omega_z \in \mathbb{R}$ is the angular rate. The robot is actuated through the wrench $\boldsymbol{u} = \begin{bmatrix}\boldsymbol{f}^\top & \tau_z\end{bmatrix}^\top \in \mathbb{R}^3$, consisting of body-frame force $\boldsymbol{f} \in \mathbb{R}^2$ and torque $\tau_z \in \mathbb{R}$. In practice, the wrench command is mapped to the robot thrusters $\mathbf{v} \in \mathbb{R}^8$ through the allocation matrix $B \in \mathbb{R}^{3\times 8}$ as $\boldsymbol{u} = B\mathbf{v}$, where B has full row rank.

The full nonlinear model of the robot is written as

$$\ddot{\boldsymbol{q}} = \bar{G}(\psi)\boldsymbol{u} \tag{1}$$

with

$$\bar{G}(\psi) = \begin{bmatrix} \frac{1}{m}R_z(\psi) & \mathbf{0}_{2\times 1} \\ \mathbf{0}_{1\times 2} & \frac{1}{I_z} \end{bmatrix} \tag{2}$$

where m is the mass of the robot, I_z is the robot inertia around the z-axis and $R_z(\psi)$ is the rotation matrix of an angle ψ about the z-axis. For convenience, we also define the state vector $\boldsymbol{x} = \begin{bmatrix}\boldsymbol{q}^\top & \dot{\boldsymbol{q}}^\top\end{bmatrix}^\top \in \mathbb{R}^5 \times (-\pi, \pi]$, such that the system dynamics can be written in input-affine form:

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}) + G(\boldsymbol{x})\boldsymbol{u} \tag{3}$$

where $f(\boldsymbol{x}) = \begin{bmatrix}\dot{\boldsymbol{q}}^\top & \mathbf{0}_{1\times 3}\end{bmatrix}^\top$ and $G(\boldsymbol{x}) = \begin{bmatrix}\mathbf{0}_{3\times 3} & \bar{G}(\psi)^\top\end{bmatrix}^\top$.

3. CONTROL METHODOLOGY

In this section, we describe the control strategy to remotely operate the ATMOS robot. Namely, the controller runs on a remote computer, which sends control inputs to the robot and receives back state feedback through the communication network. Several challenges arise from communication delays between the two sides, which we summarize as follows.

First, the robot and the controller operate on unsynchronized clocks, introducing a so-called clock bias. Secondly, the state at which a control input will be applied to the robot differs temporally from the state considered to compute such input in a feedback manner, due to transmission time delays between the robot and the controller. Moreover, the delays appear to be stochastic, some messages may be lost entirely, and the characteristics of this delay also change, albeit slowly, over time.

To address these challenges, the proposed control strategy includes a prediction step that estimates the state at which the current control inputs will be applied, representing this estimate as a distribution or belief. This predicted state is then used by a CLF-based controller, ensuring that the applied control is consistent with the estimated state.

3.1 State Prediction

Suppose, at time t_i^u on the controller clock, the controller sends a timestamped and indexed control input $\boldsymbol{u}_i$ as a packet $U_i := (\boldsymbol{u}_i, t_i^u)$. Denote by s_i^u the time on the robot clock when the i-th control packet arrives at the robot. If a packet is lost during transmission, we regard it as having arrived simultaneously with the next packet that successfully reaches the robot. For example, if the packets sent at t_i^u and t_{i+1}^u are both lost and the $(i+2)$-th packet is the first to arrive after that, we treat this as $s_i^u = s_{i+1}^u = s_{i+2}^u$. Additionally, if a packet U_i arrives later than U_{i+1} (i.e., $s_i^u \geq s_{i+1}^u$), it is discarded and considered lost, since it cannot overwrite a newer command. We assume that

[1] FleetMQ - Seamless Vehicle Middleware. `https://www.fleetmq.com/`, accessed 2025-11-24.

the control packet delay $\Delta_i = s_i^u - t_i^u$, accounting for both transmission delays and clock bias, follows a quasi-stationary distribution and that delays are uncorrelated.

At the same time, the robot sends back to the controller a state packet $X_k := (\boldsymbol{x}_k, s_k^x, i_k, \Gamma_k)$ where $\boldsymbol{x}_k = \boldsymbol{x}(s_k^x)$ is the recorded state at the time s_k^x in which the state packet is sent, $i_k \in \mathbb{N}$ is the index of the the latest applied control input $\boldsymbol{u}_{i_k}$ from packet U_{i_k} and $\Gamma_k = (\Delta_1, \dots \Delta_{N_s})$ is a continuously updated list of samples of control packet delays with $N_s > 0$. When a state packet X_k is received at the controller side, a new control packet $U_k := (\boldsymbol{u}_k, t_k^u)$ is computed, which will be applied on the robot at the unknown time s_k^u and state $\boldsymbol{x}_k^+ := \boldsymbol{x}(s_k^u)$ according to the controller shown later in (7). An estimate $\hat{\boldsymbol{x}}_k^+$ of the state $\boldsymbol{x}_k^+$ is obtained by simulating the state evolution, after the application of all the *pending* control packets $U_j,\ i_k + 1 \le j \le k$ (i.e., packets that were not yet applied before the state packet X_k was sent). Namely, for each pending packet $U_j,\ i_k + 1 \le j \le k$, the arrival time $\hat{s}_j^u$ is estimated as

$$\hat{s}_j^u = t_j^u + \hat{\Delta}_j, \tag{4}$$

where $\hat{\Delta}_j$ is randomly sampled from Γ_k. At this point, an estimate of the state $\hat{\boldsymbol{x}}_k^+$, where the currently computed packet U_k will be received and applied from the robot, is obtained assuming the zero-order hold input signal $\boldsymbol{u} : [s_k^x, \hat{s}_k^u] \to \mathbb{R}^m$ is applied on the system such that, for each interval $[\hat{s}_j^u, \hat{s}_{j+1}^u)$ with $i_k + 1 \le j < k$, it holds

$$\boldsymbol{u}(t) = \boldsymbol{u}_j,\ \forall t \in [\hat{s}_j^u, \hat{s}_{j+1}^u). \tag{5}$$

Thus $\hat{\boldsymbol{x}}_k^+$ is predicted as

$$\hat{\boldsymbol{x}}_k^+ = \hat{\boldsymbol{x}}(\hat{s}_k^u) = \int_{s_k^x}^{\hat{s}_k^u} \left(f(\boldsymbol{x}(t)) + G(\boldsymbol{x}(t))\boldsymbol{u}(t)\right) dt + \boldsymbol{x}_k. \tag{6}$$

Since the predicted arrival times $\hat{s}_j^u$ may not preserve their original order, we consider the packages such that $\hat{s}_j^u \ge \hat{s}_{j+1}^u$ as lost.

Performing the prediction in (6) for $N \in \mathbb{N}$ different samples of the delay profiles $\hat{s}_j^u$, we define a set $X_k^+ = \{\hat{\boldsymbol{x}}_{k,l}^+, \dots \hat{\boldsymbol{x}}_{k,N}^+\}$ of *particles* which approximates the predicted state distribution at which the currently generated input $\boldsymbol{u}_k$ arrives at the robot.

3.2 Reference Tracking Control

Under the given settings, it is desired to stabilize the system around a reference state $\boldsymbol{x}_{\text{ref}}$. Let $V(\boldsymbol{x}; \boldsymbol{x}_{\text{ref}})$ be a CLF designed to stabilize the original delay-free system (3) to $\boldsymbol{x}_{\text{ref}}$ i.e., $(V(\boldsymbol{x}_{\text{ref}}; \boldsymbol{x}_{\text{ref}}) = 0$ and $V(\boldsymbol{x}; \boldsymbol{x}_{\text{ref}}) > 0\ \forall \boldsymbol{x} \neq \boldsymbol{x}_{\text{ref}})$. If the state $\boldsymbol{x}_k^+$, in which the control input $\boldsymbol{u}_k$ will take effect, was known, then the system could be stabilized toward $\boldsymbol{x}_{\text{ref}}$ if $\boldsymbol{u}_k$ is such that $\dot{V}(\boldsymbol{x}_k^+, \boldsymbol{u}_k; \boldsymbol{x}_{\text{ref}}, \dot{\boldsymbol{x}}_{\text{ref}}) + \gamma V(\boldsymbol{x}_k^+; \boldsymbol{x}_{\text{ref}}) \le 0$, for a given $\gamma \in \mathbb{R}_{\ge 0}$ controlling the exponential convergence rate of the CLF. [2] Since we only have a set of particles X_k^+ to represent the true state $\boldsymbol{x}_k^+$, we propose to select $\boldsymbol{u}_k$ by solving

[2] The time derivative for $V(\boldsymbol{x}, \boldsymbol{x}_{\text{ref}})$ is given by $\dot{V}(\boldsymbol{x}, \boldsymbol{u}; \boldsymbol{x}_{\text{ref}}, \dot{\boldsymbol{x}}_{\text{ref}}) = \frac{\partial V}{\partial \boldsymbol{x}}(\boldsymbol{x}, \boldsymbol{x}_{\text{ref}})(f(\boldsymbol{x}) + G(\boldsymbol{x})\boldsymbol{u}) + \frac{\partial V}{\partial \boldsymbol{x}_{\text{ref}}}(\boldsymbol{x}, \boldsymbol{x}_{\text{ref}})\dot{\boldsymbol{x}}_{\text{ref}}$.

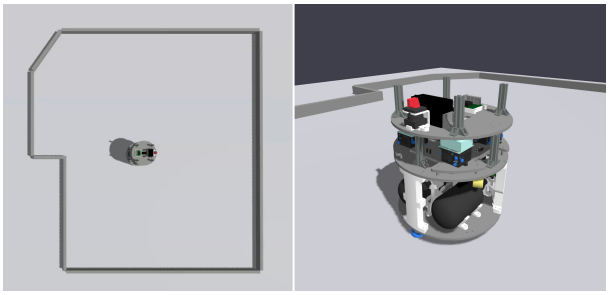


Fig. 3. The Gazebo simulator environment used in the simulation experiments, replicating the real workspace of the KTH Space Robotics Laboratory (Roque et al. (2025)).

$$\begin{aligned} \left[\boldsymbol{u}_k^T\ \delta_k\right]^T = \text{argmin}_{\boldsymbol{u},\delta} \quad & \|\boldsymbol{u} - \boldsymbol{u}_{\text{ref}}\|^2 + p\delta \\ \text{s. t. } & \delta \ge 0, \\ & \dot{V}(\hat{\boldsymbol{x}}_{k,l}^+, \boldsymbol{u}; \boldsymbol{x}_{\text{ref}}, \dot{\boldsymbol{x}}_{\text{ref}}) + \gamma V(\hat{\boldsymbol{x}}_{k,l}^+; \boldsymbol{x}_{\text{ref}}) \le \delta, \\ & \qquad \forall \hat{\boldsymbol{x}}_{k,l} \in X_k^+ \end{aligned} \tag{7}$$

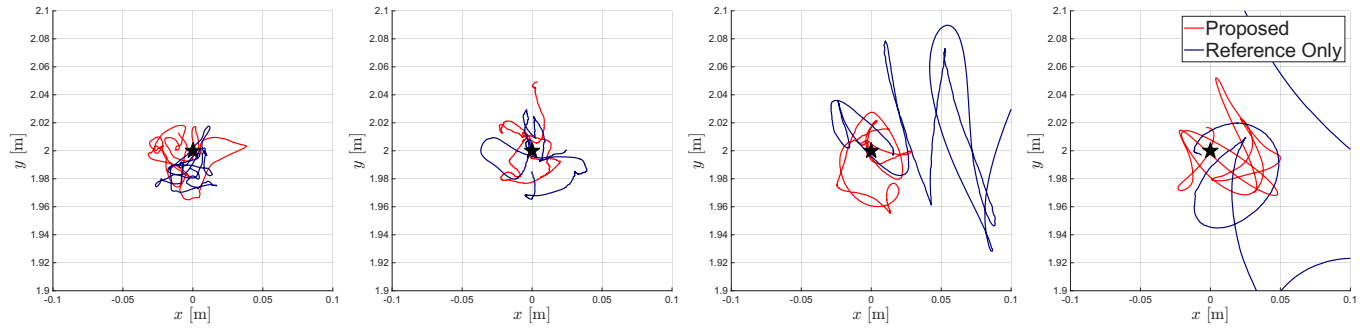
where δ is a slack variable that relaxes the CLF decrease condition when strict satisfaction is not possible for all predicted states; $p \in \mathbb{R}_{\ge 0}$ is a penalty weight, $\dot{\boldsymbol{x}}_{\text{ref}}$ is the state velocity of the reference trajectory, $\boldsymbol{u}_{\text{ref}}$ is the reference input typically deriving from a stabilizing controller or a manual control signal from an operator.

The optimization-based controller (7) tries to search for the input $\boldsymbol{u}_k$ such that 1) $\boldsymbol{u}_k$ minimally deviates from the reference $\boldsymbol{u}_{\text{ref}}$ and 2) $\boldsymbol{u}_k$ enforces an exponential decrease of the CLF (or a minimal increase thereof), for all possible predicted states $\hat{\boldsymbol{x}}_{k,l}^+ \in X_k^+$. These two potentially conflicting objectives are balanced via the positive tunable parameter p, where a bigger p implies a priority on the satisfaction of the CLF constraint, resulting in a more robust but conservative behavior.

Considering the input-affine dynamics in (3), the optimization in (7) is a standard quadratic program with linear constraints, which can be solved efficiently, and is always feasible. While formal guarantees on the stability of the system under (7) with stochastic delays are difficult and omitted here, the fact that the CLF constraint $\dot{V} + \gamma V \le \delta$ is enforced for all predicted states $\hat{\boldsymbol{x}}_{k,l}^+ \in X_k^+$ naturally yields a worst-case-aware controller that provides some robustness toward the assumption considered over the time delay distribution. This robustness property was indeed confirmed in simulation and hardware experiments.

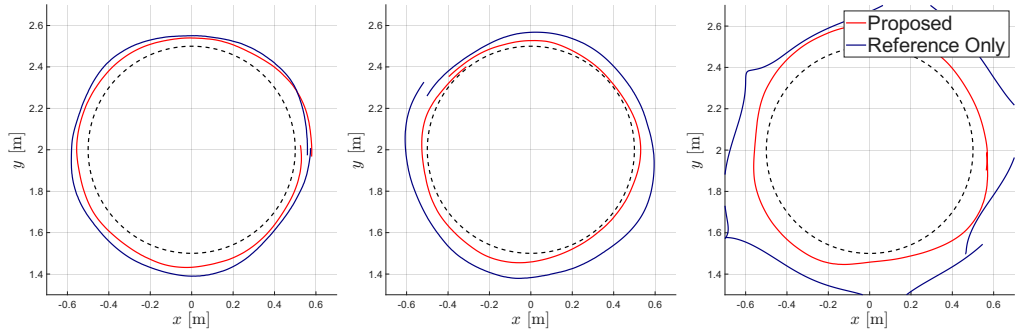
4. SIMULATION EXPERIMENTS

To showcase the validity of the proposed control method against delayed network, we first present Software- In-The-Loop (SITL) simulation results using simulated stochastic delays. Compared to the hardware experiments, simulation experiments, where one is free to modify the delay characteristics, allow to analyze more on how the control performance varies with respect to different types of delays. The simulation was done on a desktop computer equipped with a 2.40 GHz CPU and 32 GB RAM, running ROS 2, Gazebo simulator (Fig. 3) based on Roque et al. (2025) and its accompanying software, and CVXOPT as the QP solver to solve (7).



(a) No delay (b) Light delay (c) Moderate delay (d) Severe delay

Fig. 4. Position trajectories under various delay levels with and without delay compensation in stabilization task of the simulation experiment. The ★ symbols denote the stabilization target positions.



(a) No delay (b) Light delay (c) Moderate delay

Fig. 5. Position trajectories under various delay levels with and without delay compensation in the tracking task of the simulation experiment. The reference circular trajectory is drawn as black dotted curves.

Table 1. Delay characteristics used in the simulation experiments

	a [ms]	b [ms]	Drop rate [%]
No delay	0	0	0
Light delay	100	200	20
Moderate delay	200	400	20
Severe delay	300	600	20

In the simulation experiment, we consider two tasks. The *stabilization* task is to stabilize the robot to a fixed point. In the *tracking* task, the robot is required to track a circular trajectory of 1-meter diameter, facing the center of the circle. The speed along the trajectory remains constant, and one revolution corresponds to 30 seconds. For each task, four different delay characteristics were considered, which simulate the delay according to

$$\Delta_i = s_i^u - t_i^u = \mathrm{clip}(a + b \cdot \mathcal{U}_{[0,1]}^i, [a, b]), \tag{8}$$

where $a > 0$ and $b > 0$ are parameters controlling the severity of delay, the function clip clips the first argument to fit in the specified interval, and $\mathcal{U}_{[0,1]}^i$-s are independent and identically distributed (i.i.d.) samples from a uniform distribution over the interval $[0, 1]$. In addition to this randomness, we regard part of the messages as lost, according to a constant drop rate. The parameter values for the delay profiles considered in the simulations are summarized in Table 1, ranging from no delay to severe delay. In all tasks, the reference input $\boldsymbol{u}_{\mathrm{ref}}$ is selected as a well-tuned proportional-derivative (PD) reference tracking controller. The CLF was designed using the linear quadratic regulator (LQR) technique with respect to the linearized dynamics.

In Table 2, we evaluate the performance of the proposed control strategy and compare with the naive approach where only the reference PD controller was used. The evaluation and comparison are done with respect to the root mean square error (RMSE) of position and yaw angles. In Fig. 4 and Fig. 5, we depict the position trajectories over thirty-second intervals. The severe delay case of the tracking task was omitted, where both controllers failed to stabilize to the reference trajectory, and thus, the RMSE metric and the trajectories lost significance.

As expected, in both controllers, the tracking error increases as the delay severity escalates. It is also a clear trend in the recorded data that the proposed method provides a more stable maneuver of the robot with significantly reduced position and orientation tracking errors, both in the stabilization and tracking tasks. It is also notable that the difference between the two controllers is not significant where delay is not severe, because the controller (7) was designed to not overly interfere with the reference controller if $\boldsymbol{u}_{\mathrm{ref}}$ is already a *sufficiently stable* input that decreases the CLF value for all predicted state in X^+. In such cases where the effect of delays is not significant, the state predictions in X^+ shows high accuracy, and the PD reference controller $\boldsymbol{u}_{\mathrm{ref}}$ is already likely to be a sufficiently stable input for all particles in X^+, achieving $\delta = 0$ in (7).

5. HARDWARE DEMONSTRATION

For hardware demonstration, we consider a docking task in which the ATMOS robot is operated under remote closed-loop control over the internet to dock to a docking station

Table 2. Simulation result: Position and yaw RMSE under various delay levels

	Stabilization Task				*Tracking* Task			
	Proposed		Reference Only		**Proposed**		Reference Only	
	Position [m]	Yaw [°]	Position [m]	Yaw [°]	Position [m]	Yaw [°]	Position [m]	Yaw [°]
No delay	0.022	0.536	0.020	0.430	0.069	0.767	0.081	8.511
Light delay	0.022	1.515	0.025	1.714	0.060	1.496	0.118	15.456
Moderate delay	0.024	2.385	0.212	31.977	0.104	2.318	0.326	91.094
Severe delay	0.032	4.421	0.618	76.208	-	-	-	-

installed on the boundary of the low-friction experiment arena, as shown in Fig. 1.

5.1 Setup

In the experiment setup, the ATMOS robot in Stockholm, Sweden, is controlled from the controller which is physically located in Seoul, Korea. This is approximately 7450 km away from the robot, as shown in Fig. 2.

To convey the control and feedback messages over the network, we use FleetMQ as the middleware. FleetMQ is a dynamic middleware platform designed for real-time communication between distributed robotic systems operating across heterogeneous networks. FleetMQ automatically establishes the shortest peer-to-peer network route between the controller and the robot.

The experiment is conducted on a planar microgravity-emulating testbed, where the free-flying ATMOS robot moves on a flat low-friction arena. The platform floats on three passive air-bearings, rendering its in-plane dynamics effectively weightless. The robot is actuated via eight solenoid air-valves at 10 Hz and is equipped with a rigid conic anchoring tool for docking with a mating passive docking station placed at the edge of the arena.

Robot motion is tracked using a high-frequency Qualisys motion-capture system. The pose measurements are fused onboard with IMU data, producing a state estimate transmitted to the controller. More details are provided in Roque et al. (2025).

5.2 Motion Planning

The installed docking station and the mating connectors requires the robot to be controlled at a precision of less than a few centimeters for a successful docking. Thus, we employ the following two-step strategy to generate the docking trajectory, which is used as the reference $\boldsymbol{x}_{\text{ref}}$ that is fed into the CLF-based controller (7).

(1) The robot stabilizes to the *parking configuration*. The parking configuration is set as directly facing the docking station in its front. In this step, the reference state $\boldsymbol{x}_{\text{ref}}$ is the same as the parking configuration with zeros in the velocity components.
(2) When the robot approaches sufficiently close to the parking configuration, an optimal control problem is solved online to generate a feasible docking trajectory and the corresponding reference for the CLF controller.

In this demonstration, we set the parking position to be 50 cm away from the docking station. The optimal control problem in the second step is formulated as follows. First, the reference yaw angle ψ_d is always set to face the docking

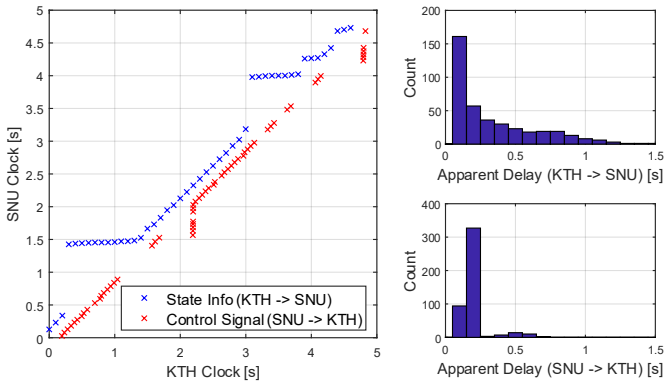


Fig. 6. (Left) The message transmission and arrival times. Each point corresponds to one message, whose horizontal and vertical coordinates representing the time when it first appeared on the KTH and the SNU sides, respectively. (Right) The histograms of the apparent transmission delay. Since the two clocks cannot (and need not) be perfectly synchronized, the one-way delay is not accurately observable and the apparent delay might be biased.

position throughout the trajectory and is not part of the optimization. We solve the following optimization to solve for the desired positions and linear velocities as a function of time:

$$\min_{\boldsymbol{p}_{(\cdot)},\boldsymbol{v}_{(\cdot)},\boldsymbol{a}_{(\cdot)}} \quad \sum_{k=0}^{K} \boldsymbol{a}_k^\top \boldsymbol{Q}_{\boldsymbol{k}} \boldsymbol{a}_k \tag{9a}$$

$$\text{s.t.} \quad \begin{bmatrix} \boldsymbol{p}_{k+1} \\ \boldsymbol{v}_{k+1} \end{bmatrix} = A \begin{bmatrix} \boldsymbol{p}_k \\ \boldsymbol{v}_k \end{bmatrix} + B\boldsymbol{a}_k, \tag{9b}$$

$$\begin{bmatrix} \boldsymbol{p}_K \\ \boldsymbol{v}_K \end{bmatrix} = \begin{bmatrix} \boldsymbol{p}_{\text{dock}} \\ \boldsymbol{0}_{3\times 1} \end{bmatrix}, \tag{9c}$$

$$-\boldsymbol{a}_{\max,k} \le \boldsymbol{a}_k \le \boldsymbol{a}_{\max,k}, \tag{9d}$$

where $k \in [0, K]$ is the time index, $\boldsymbol{p}_k$, $\boldsymbol{v}_k$ and $\boldsymbol{a}_k$ are respectively the position, linear velocity, and acceleration of the trajectory, A and B denote the discrete-time translational dynamics (in the form of a double integrator model) of the robot. The optimization is done subject to the docking (9c) and the acceleration limit (9d) constraints. The acceleration limit $\boldsymbol{a}_{\max,k} \ge 0$ is set to be a decreasing function with respect to k to reduce the effect of possible deviation from the docking trajectory caused by applying high accelerations.

5.3 Results

Fig. 6 depicts the transmission delay experienced by both sides during the experiment. The actual delay was normally constant, lying between 100 ms and 200 ms (which is about the time required to make a round-trip between the

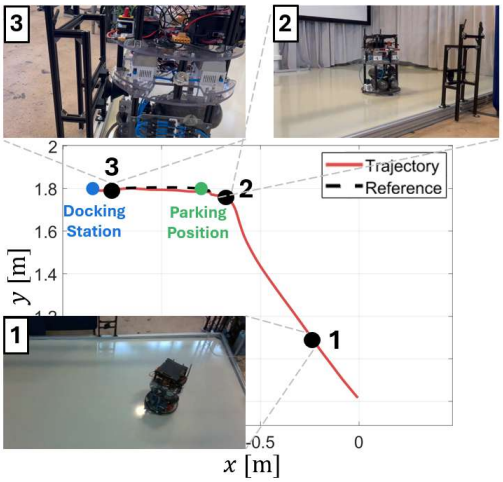


Fig. 7. The position trajectory of the hardware experiment. The robot first stabilizes to the docking configuration marked green, and then the optimal trajectory computed from the optimal control problem (9) is executed.

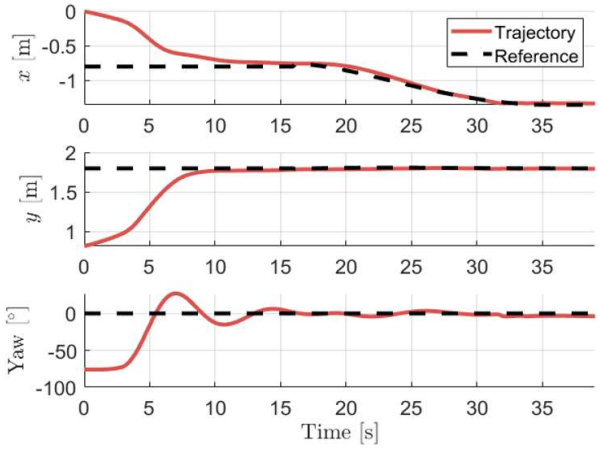


Fig. 8. The configuration trajectory recorded in the hardware experiment. Even under losses of data and transmission delays, the actual robot trajectory well converged to the desired trajectory.

two sides in the speed of light), with intermittent blockages of up to 1 second long every 3 to 5 seconds. Here, the source of transmission delay is not only the physical time required for the information to travel, but also the bottleneck effect due to limited network bandwidth, etc. Indeed, this delay is highly time-correlated, and thus, the assumptions made in the belief prediction step do not hold. However, as seen in Fig. 7 and Fig. 8, the proposed control strategy showed satisfactory performance and successfully accomplished the task, proving the robustness of the approach.

6. CONCLUSION

In this work, we demonstrated a remotely controlled docking task of the ATMOS robot, a hardware platform for research in space robotics. We develop a remote control scheme that combines state prediction against the non-stationary and stochastic transmission delay with a CLF-based controller that mitigates the effects of uncertainties and possible distribution mismatches deriving from the stochastic nature of the experimental setting. Our demonstration successfully showcased that the proposed control scheme enables stable operation of the ATMOS robot over a 7450 km distance between Seoul (controller location) and Stockholm (ATMOS location).

We conclude by suggesting possible future research directions. First, although empirical results suggest that the proposed control strategy enables stable remote control across a long distance, theoretical guarantees of performance and stability will be studied to improve the reliability and, therefore, practical applicability of the controller for space robotics tasks. Another important future research direction is to minimize the amount of information exchange, possibly by combining event-based control methods such as Gu et al. (2022), as the stochasticity in transmission delay often results from the saturation of network bandwidth.

REFERENCES

Chen, H. and Liu, Z. (2021). Time-delay prediction–based smith predictive control for space teleoperation. *Journal of Guidance, Control, and Dynamics*, 44(4), 872–879.

Freeman, R.A. and Primbs, J.A. (1996). Control Lyapunov functions: New ideas from an old source. In *Proceedings of 35th IEEE conference on decision and control*, volume 4, 3926–3931. IEEE.

Gu, Z., Yan, S., Ahn, C.K., Yue, D., and Xie, X. (2022). Event-triggered dissipative tracking control of networked control systems with distributed communication delay. *IEEE Systems Journal*, 16(2), 3320–3330. doi:10.1109/JSYST.2021.3079460.

Liu, X. and Kumar, K.D. (2012). Network-based tracking control of spacecraft formation flying with communication delays. *IEEE Transactions on Aerospace and Electronic Systems*, 48(3), 2302–2314.

Roque, P., Phodapol, S., Krantz, E., Lim, J., Verhagen, J., Jiang, F.J., Dörner, D., Mao, H., Tibert, G., Siegwart, R., Stenius, I., Tumova, J., Fuglesang, C., and Dimarogonas, D.V. (2025). Towards open-source and modular space systems with atmos. *IEEE Transactions on Field Robotics*. doi:10.1109/TFR.2025.3632772.

Sheridan, T. (1993). Space teleoperation through time delay: review and prognosis. *IEEE Transactions on Robotics and Automation*, 9(5), 592–606. doi:10.1109/70.258052.

Torgerson, J.L., Cerf, V., DeBaun, S.U., and Suzuki, L.C. (2024). Space system internetworking: The foundational role of delay and disruption-tolerant networking. *IEEE Journal on Selected Areas in Communications*, 42(5), 1359–1370.

Wang, W., Li, C., Sun, Y., and Ma, G. (2019). Distributed coordinated attitude tracking control for spacecraft formation with communication delays. *ISA transactions*, 85, 97–106.

Zhang, J., Xia, H., and Ma, G. (2024). Fixed-time coordinated attitude control for spacecraft formation flying via event-triggered and delayed communication. *Advances in Space Research*, 73(12), 6140–6160.

Zhang, Z., Shi, Y., Zhang, Z., Zhang, H., and Bi, S. (2018). Modified order-reduction method for distributed control of multi-spacecraft networks with time-varying delays. *IEEE Transactions on Control of Network Systems*, 5(1), 79–92. doi:10.1109/TCNS.2016.2578046.