PX4Space: PX4 for Spacecraft and Space Robotics

Pedro Roque[1], Elias Krantz[2], Jaeyoung Lim[3], Christer Fuglesang[2], Roland Siegwart[3], Dimos V. Dimarogonas[1]

***Abstract*— With the rise of interest in space robotics, both from an academic and industrial point of view, the need for standardization of testing facilities is ever greater. Providing an open-source solution that multiple parties can use and share improvements on becomes pivotal for supporting research and technology transfer roles, accelerating the progress in autonomy for this field. In this short note, we present an adaptation of the PX4 software stack for usage with spacecraft analogs and space robotics facilities. We showcase the proposed architecture, controllers, and interfaces both in simulation and at the space robotics laboratory in KTH.**

I. INTRODUCTION

Space robotics has recently become popular to augment space systems with recent progress in decision-making capabilities developed for terrestrial systems. Due to the challenges in sending hardware to orbital environments, ground experiment testbeds such as CAST Spacecraft Simulator [1], ESTEC's Orbital Robotics Lab [2], and SNU's Zero-G laboratory [3] are vital for space research. However, conducting experiments is still expensive and challenging due to the use of custom-built software and inaccessible hardware.

Providing an open-source solution for space robotics would be pivotal for supporting research and technology transfer roles. MIT SPHERES [4] and NASA Astrobee [5] have paved the way for the space robotics community to make space-deployed hardware more accessible, anchored by open-source software and simulation tools. However, these software are developed for a specific hardware which doesn't scale to other platforms.

Generic open-source hardware [6] and software [7] have had a major impact on accelerating aerial robotics research. Such shared hardware and software infrastructure within the aerial robotics community has allowed researchers to focus on the key research problems.

In this work, we present an adaptation of the PX4 Autopilot [7] for space robotics. We take advantage of the generic architecture of PX4, to create a generic hardware-agnostic software framework for space robotics. We showcase the proposed framework in a software-in-the-loop (SITL) simulation which can be easily transitioned to a hardware platform available at the Space Robotics Laboratory[1] in KTH, Sweden. Our key contributions are i) addition of interfaces for thruster-based control allocation; ii) custom modules for space robotics; iii) a development environment using SITL for testing control systems; and iv) demonstration on 2D spacecraft hardware. The remainder of the paper is organized as follows: an overview of the system architecture is provided in Section II; results are shown in Section III; and concluding remarks are discussed in Section IV.

II. ARCHITECTURE

In this section we provide an overview of the developed interfaces. The source code of PX4Space, along with an accompanying QGroundControl interface, are available at:
`https://github.com/DISCOWER/PX4-Space-Systems`
`https://github.com/DISCOWER/qgroundcontrol`

A. Control

We show the control scheme implemented in the *PX4Space* Firmware as shown in Fig. 2. The control system is composed of three cascaded PI and PID controllers. The cascaded loop structure allows using *PX4Space* as a low-level controller of multiple levels to evaluate different control methods. The Position controller receives a position setpoint $\boldsymbol{p}$ and regulates it through a PI controller, generating an internal velocity setpoint tracked by a PID for the velocity error. An attitude setpoint $\boldsymbol{\beta} = \{\mathbf{f}_{ref}, \mathbf{q}_{ref}\}$ is generated, corresponding to a body-frame thrust and a target attitude setpoint, respectively. The attitude controller implements a P-controller to regulate the attitude reference, generating an angular rate setpoint $\boldsymbol{\gamma} = \{\mathbf{f}_{ref}, \boldsymbol{\omega}_{ref}\}$. Lastly, the rate controller generates a wrench setpoint $\delta = \{\mathbf{f}_{ref}, \tau_{ref}\}$, through a PID to converge the angular velocity to the target.

To actuate each thruster on the platform, the wrench setpoint $\boldsymbol{\delta}$ is processed by the control allocator (CA) module. The platform geometry is defined by configuring the location of each thruster, along with its effectiveness. For details, we refer the reader to the airframe `70000_spacecraft_2d` in the codebase. The CA then sends a normalized thrust to each actuator i, $\boldsymbol{f}_i \in [0, 1]$, as a PWM signal.

It is worth noting that the above setpoints are specific to the implementation of the cascaded loop structure. Due to the modular structure of PX4, it is easy to add a custom module. Moreover, these setpoints can be generated not only from

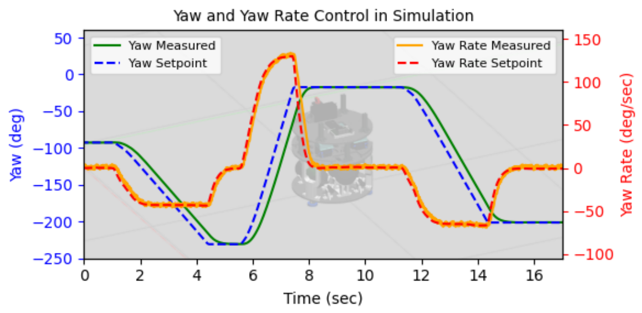
[1] Pedro Roque and Dimos V. Dimarogonas are with the Division of Decision and Control Systems, KTH Royal Institute of Technology, Stockholm, Sweden. {padr, dimos}@kth.se.

[2] Elias Krantz and Christer Fuglesang are with the Department of Engineering Mechanics, KTH Royal Institute of Technology, Stockholm, Sweden. {eliaskra, cfug}@kth.se.

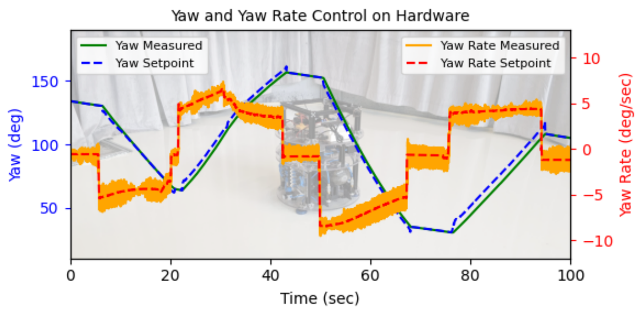
[3]Autonomous Systems Lab, ETH Zürich, Switzerland. {jalim, rsiegwart}@ethz.ch

This work was supported by the H2020 ERC Grant LEAFHOUND, the Swedish Research Council (VR), the Knut och Alice Wallenberg Foundation (KAW), and the Wallenberg AI, Autonomous Systems and Software Program (WASP) DISCOWER funded by KAW.

[1]Website: www.discower.io, accessed on 29th of May, 2024.



(a) SITL performance. In the background, the simulated platform.



(b) Hardware test. In the background, the lab platform used.

Fig. 1: Attitude control performance in SITL and on 2D spacecraft hardware in KTH.

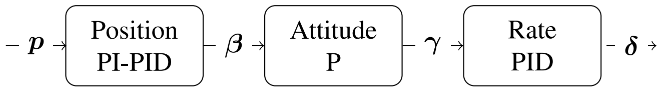


Fig. 2: Cascaded control scheme implemented in *PX4Space*. The references $\boldsymbol{p}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\delta}$ represent position, attitude, rate, and thrust/torque setpoints, respectively.

a module or through manual inputs, but also via external entities such as an onboard computer.

B. *Software-in-the-Loop Simulation*

The SITL simulation environment utilizes the POSIX compatibility of PX4, where the firmware can be simulated directly on any POSIX environment. This allows running the firmware directly on the laptop allowing to easily and accurately simulate the firmware as it would be deployed on real hardware. We use Gazebo-Classic [8] as the dynamics simulator, where PX4 communicates with the simulator via the MAVLink protocol. The simulated gas thrusters[2] are a latching force model given by $\mathbf{f}_{thrust} = \mathbf{f}_{max}\lambda$ where, λ is 1 during the *on* part of the PWM duty cycle.

III. EVALUATION

We present preliminary results both in SITL and Hardware for our attitude control modules. We encourage the reader to test these and more modules on the provided codebase.

The target platform is a 2D spacecraft available at KTH Space Robotics Laboratory. The system weighs 12.5 kg with dimension of 40 cm in diameter and 50 cm tall. The system has eight cold gas thrusters, which use compressed air to generate thrust outputting a maximum of 1.4 N. The avionics consists of a flight management unit (FMU) and a mission computer. The FMU executes the low-level controls and communicates with ROS 2 over the XRCE-DDS bridge. This allows communicating directly through the internal messaging system of PX4. Local position is provided through a Qualisys motion capture system to the hardware platform. On the hardware platform, the vehicle is equipped with a Pixhawk V6X FMU and a Jetson Orin for the mission computer. A simulated version of the platform was created in Gazebo-Classic.

[2]Available at: `https://github.com/PX4/PX4-SITL_gazebo-classic/tree/pr-thruster-plugin`

Manual setpoints are provided to the platform both in simulation and in hardware. The performance of the system is shown in Fig. 1, with dashed lines representing setpoints and solid lines the measured state. The setpoints are appropriately tracked during the entire motion.

IV. CONCLUSIONS AND FUTURE WORK

In this work we proposed an adaptation of PX4 Autopilot project, *PX4Space*, for space robotics research in a laboratory environment. The solution enables testing in SITL simulation and a seamless transition to hardware testing. Preliminary results show the proposed software in two different control modes. For future work, compatibility with other infrastructure in the PX4 ecosystem, such as QGroundControl would provide even more opportunities for development.

REFERENCES

[1] Y. K. Nakka, R. C. Foust, E. S. Lupu, D. B. Elliott, I. S. Crowell, S.-J. Chung, and F. Y. Hadaegh, “A six degree-of-freedom spacecraft dynamics simulator for formation control research,” in *2018 AAS/AIAA Astrodynamics Specialist Conference*, 2018, pp. 1–20.

[2] H. Kolvenbach and K. Wormnes, “Recent developments on orbit, a 3-dof free floating contact dynamics testbed,” in *13th International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS 2016)*, vol. 6, 2016.

[3] M. Olivares-Mendez, M. R. Makhdoomi, B. C. Yalçın, Z. Bokal, V. Muralidharan, M. O. Del Castillo, V. Gaudilliere, L. Pauly, O. Borgue, M. Alandihallaj *et al.*, “Zero-g lab: a multi-purpose facility for emulating space operations,” *Journal of Space Safety Engineering*, vol. 10, no. 4, pp. 509–521, 2023.

[4] D. Miller, A. Saenz-Otero, J. Wertz, A. Chen, G. Berkowski, C. Brodel, S. Carlson, D. Carpenter, S. Chen, S. Cheng *et al.*, “Spheres: a testbed for long duration satellite formation flying in micro-gravity conditions,” in *Proceedings of the AAS/AIAA space flight mechanics meeting*, vol. 105. Citeseer, 2000, pp. 167–179.

[5] M. Bualat, J. Barlow, T. Fong, C. Provencher, and T. Smith, “Astrobee: Developing a free-flying robot for the international space station,” in *AIAA SPACE 2015 conference and exposition*, 2015, p. 4643.

[6] L. Meier, P. Tanskanen, F. Fraundorfer, and M. Pollefeys, “Pixhawk: A system for autonomous flight using onboard computer vision,” in *2011 ieee international conference on robotics and automation*. IEEE, 2011, pp. 2992–2997.

[7] L. Meier, D. Honegger, and M. Pollefeys, “Px4: A node-based multithreaded open source robotics framework for deeply embedded platforms,” in *2015 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2015, pp. 6235–6240.

[8] N. Koenig and A. Howard, “Design and use paradigms for gazebo, an open-source multi-robot simulator,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sendai, Japan, Sep 2004, pp. 2149–2154.