doi: 10.82124/CEAS-GNC-2026-086 CEAS-GNC-2026-086

Mixed-Integer vs. Continuous Model Predictive Control for Binary Thrusters: A Comparative Study

Franek Stark Robotics Researcher, Robotics Innovation Center, German Research Center for Artificial Intelligence GmbH, Bremen, Germany. franek.stark@dfki.de

Jakob Middelberg Research Assistant, Robotics Innovation Center, German Research Center for Artificial Intelligence GmbH, Bremen, Germany. jakob.middelberg@dfki.de
Master Student, University of Bremen, Germany. middelberg@uni-bremen.de

Shubham Vyas Robotics Researcher, Robotics Innovation Center, German Research Center for Artificial Intelligence GmbH, Bremen, Germany. shubham.vyas@dfki.de

ABSTRACT

Binary on/off thrusters are commonly used for spacecraft attitude and position control during proximity operations. However, their discrete nature poses challenges for conventional continuous control methods. The control of these discrete actuators is either explicitly formulated as a mixed-integer optimization problem or handled in a two-layer approach, where a continuous controller's output is converted to binary commands using analog-to-digital modulation techniques such as Delta-Sigma ($\Delta\Sigma$)-modulation. This paper provides the first systematic comparison between these two paradigms for binary thruster control, contrasting continuous Model Predictive Control (MPC) with Delta-Sigma modulation against direct Mixed-Integer MPC (MIMPC) approaches. Furthermore, we propose a new variant of MPC for binary actuated systems, which is informed using the state of the Delta-Sigma Modulator. The two variations for the continuous MPC along with the MIMPC are evaluated through extensive simulations using ESA's REcap-ACrobat-SAtsim (REACSA) platform. Results demonstrate that while all approaches perform similarly in high-thrust regimes, MIMPC achieves superior fuel efficiency in low-thrust conditions. Continuous MPC with modulation shows instabilities at higher thrust levels, while binary informed MPC, which incorporates modulator dynamics, improves robustness and reduces the efficiency gap to the MIMPC. It can be seen from the simulated and real-system experiments that MIMPC offers complete stability and fuel efficiency benefits, particularly for resource-constrained missions, while continuous control methods remain attractive for computationally limited applications.

Keywords: model predictive control, mixed-integer optimization, satellite control

Nomenclature

θ	=	Platform orientation (yaw) in world coordinates
$\dot{\theta}$	=	Platform angular velocity in world coordinates
x, y	=	Platform position in world coordinates
$\dot{x}, \dot{y}$	=	Platform velocity in world coordinates
ω_{RW}	=	Reaction Wheel (RW) rotational speed

s_θ, c_θ	=	Sine and cosine of θ
$\mathbf{x}, \mathbf{u}$	=	State and input vector of the system
$\hat{\mathbf{x}}$	=	Target state
F_n	=	Nominal thrust force applied by a single thruster
r	=	Platform radius
m, I_S	=	Platform overall mass and inertia on the z-axis.
I_RW	=	RW's inertia on the z-axis.
$\mathbf{x}_{j\|t}, \mathbf{u}_{j\|t}$	=	State and input prediction of time step j, predicted at time step t
N	=	Prediction Horizon
Δt	=	Discretization time of the system dynamics within the controller
$\mathbf{U}_t, \mathbf{X}_t$	=	Set of all predicted states and inputs at time step t
$\mathbf{u}_{\mathrm{bin},t}$	=	Set of all predicted binary inputs at time step t
$\mathbf{Q}, \mathbf{W}$	=	Diagonal cost matrices for state and input cost terms.

1 Introduction

Satellites and space vehicles are commonly equipped with either cold gas or hypergolic thrusters for attitude and position control during proximity operations [1]. These thrusters are usually binary actuated, i.e., they are on/off devices and can only be either fully on or fully off. This is due to the fact that the thrusters are ordinarily designed to operate at a specific pressure and flow rate, and operating them at partial thrust can lead to instability and inefficiency. Furthermore, on/off thrusters are often simpler than variable-thrust thrusters, leading to higher reliability and lower cost. However, most control methods assume a smooth and continuous state and control input, which is not given with on/off thrusters. Therefore, many existing control methods outlined in the literature are not directly applicable, making it challenging to achieve precise attitude and position control. This results in two options for the development of control systems for proximity operations with binary thrusters. The first is to use a two-layer control architecture, where a continuous controller generates the desired control inputs, and a separate module converts these inputs into binary commands for the thrusters. Different methods have been proposed, from using simple thresholding logic [2], up to modulators like pulse-width modulation (PWM) [3], $\Delta\Sigma$-modulation [4, 5] or similar schemes to convert analog/continuous control input to digital/discrete values. The second option is to directly perform control on the binary control inputs using Mixed Integer (MI) methods. Until recently, these were considered too computationally expensive for real-time control, but recent advances in MI optimization have made it possible to solve these problems in real-time [6–9]. The first approach, while computationally advantageous, can lead to suboptimal performance, as the continuous controller may not be able to fully account for the limitations of the binary thrusters, such as dwell-time constraints (ramp-up or cool-down time). The second approach, while more complex, can lead to better performance, as it can directly account for the limitations of the binary thrusters. While earlier work has compared the performance of PWM and $\Delta\Sigma$-modulation [4], to the best of the authors' knowledge, there has been no work comparing the performance of the two-layer approach with MI methods for controlling systems with binary thrusters. Such a comparison would allow for understanding the trade-offs between the two approaches and help in understanding if the increased effort for MI methods is justified by the performance gains.

In this paper, we compare the performance of a two-layer control architecture using a continuous input Model Predictive Control (MPC) with $\Delta\Sigma$-modulation with a Mixed Integer Model Predictive Control (MIMPC) approach for controlling a system with on/off thrusters. We also introduce a new MPC approach which uses a continuous input MPC, that is informed about the behavior of the $\Delta\Sigma$-modulator in the prediction horizon. This allows the MPC to account for the limitations of the binary thrusters while still being able to use efficient numerical methods for solving the continuous optimization problem. We compare the performance of these three approaches in terms of their ability to reach and maintain

a target position, as well as their efficiency in terms of thruster usage. The comparison is performed through a simulation study along with experimental validation of the newly proposed binary-informed MPC approach at the Orbital Robotics & GNC Lab (ORGL) of the European Space Agency (ESA) [10]. Both the simulation and hardware datasets[1], together with the controller implementation[2], are publicly available as open source resources.

2 System Description

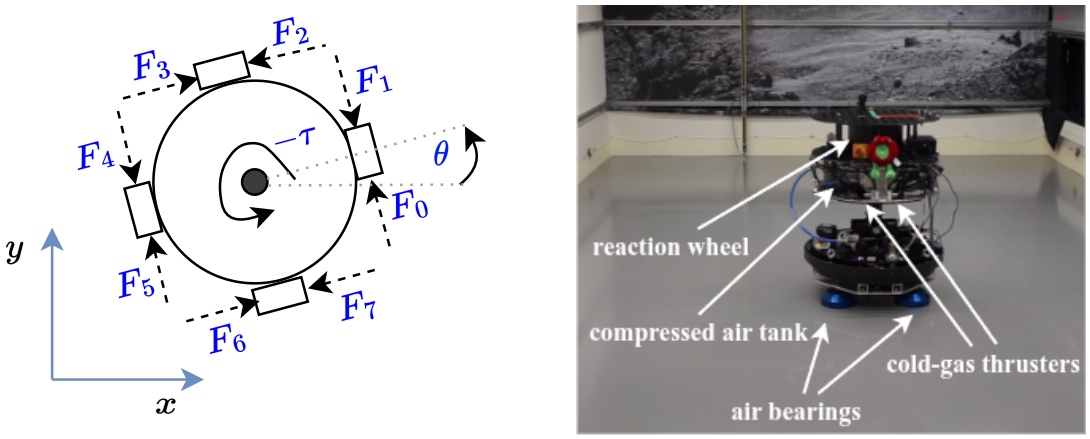

Fig. 1 Sketch (left) of the free-floating platform REcap-ACrobat-SAtsim (REACSA) (right)

The system, which serves as an example for a satellite or small spacecraft, is ESA's free-floating platform REACSA [11]. It is a 200 kg heavy air-bearing platform with a propulsion system containing eight cold-gas thrusters and an additional Reaction Wheel (RW). A sketch of the system, showing the thruster configuration, is shown along with a picture in Figure 1. The cold-gas propulsion enforces a minimum thruster activation time $t_{\text{on, min}} = 0.1\,\text{s}$ and limits the maximum firing time of a single thruster to $t_{\text{on, max}} = 0.3\,\text{s}$. Further, it enforces a cool-down time of $t_{\text{off, min}} = 0.2\,\text{s}$ between consecutive firings of the same thrusters. REACSA's overall system properties are given in Table 1. REACSA is modeled as a linear time-varying system, the dynamics are given by:

$$\dot{\mathbf{x}} = \underbrace{\begin{bmatrix} \mathbf{0}^{3\times3} & \mathbf{I}^{3\times3} & \mathbf{0}^{3\times1} \\ & \mathbf{0}^{4\times7} & \end{bmatrix}}_{\mathbf{A}} \mathbf{x} + \underbrace{\begin{bmatrix} & \mathbf{0}^{3\times9} & \\ 0 & -\mathrm{s}_\theta \frac{F_\mathrm{n}}{m} & \mathrm{s}_\theta \frac{F_\mathrm{n}}{m} & -\mathrm{c}_\theta \frac{F_\mathrm{n}}{m} & \mathrm{c}_\theta \frac{F_\mathrm{n}}{m} & \mathrm{s}_\theta \frac{F_\mathrm{n}}{m} & -\mathrm{s}_\theta \frac{F_\mathrm{n}}{m} & \mathrm{c}_\theta \frac{F_\mathrm{n}}{m} & -\mathrm{c}_\theta \frac{F_\mathrm{n}}{m} \\ 0 & \mathrm{c}_\theta \frac{F_\mathrm{n}}{m} & -\mathrm{c}_\theta \frac{F_\mathrm{n}}{m} & -\mathrm{s}_\theta \frac{F_\mathrm{n}}{m} & \mathrm{s}_\theta \frac{F_\mathrm{n}}{m} & -\mathrm{c}_\theta \frac{F_\mathrm{n}}{m} & \mathrm{c}_\theta \frac{F_\mathrm{n}}{m} & \mathrm{s}_\theta \frac{F_\mathrm{n}}{m} & -\mathrm{s}_\theta \frac{F_\mathrm{n}}{m} \\ \frac{-1}{I_\mathrm{S}} & \frac{F_\mathrm{n} r}{I_\mathrm{S}} & -\frac{F_\mathrm{n} r}{I_\mathrm{S}} & \frac{F_\mathrm{n} r}{I_\mathrm{S}} & -\frac{F_\mathrm{n} r}{I_\mathrm{S}} & \frac{F_\mathrm{n} r}{I_\mathrm{S}} & -\frac{F_\mathrm{n} r}{I_\mathrm{S}} & \frac{F_\mathrm{n} r}{I_\mathrm{S}} & -\frac{F_\mathrm{n} r}{I_\mathrm{S}} \\ \frac{1}{I_\mathrm{RW}} & & & \mathbf{0}^{1\times8} & & & & & \end{bmatrix}}_{\mathbf{B}(\theta)} \mathbf{u} \quad (1)$$

with the state vector $\mathbf{x} = \begin{bmatrix} x & y & \theta & \dot{x} & \dot{y} & \dot{\theta} & \omega_\mathrm{RW} \end{bmatrix}^T$ containing the system's pose, velocity and RW velocity. The input vector is $\mathbf{u}_\mathrm{bin} = \begin{bmatrix} u_0 & u_1 & u_2 & u_3 & u_4 & u_5 & u_6 & u_7 & u_8 \end{bmatrix}^T$, where the first element is the RW's continuous torque $u_0 \in \mathbb{R}$. The remaining eight elements are the binary thruster activations $u_{1\ldots8} \in \{0, 1\}$. An important fact is that systems with on/off thrusters and a minimum firing time, also

[1]Datasets: `https://doi.org/10.5281/zenodo.18454916`
[2]Controller implementation: `https://github.com/dfki-ric-underactuated-lab/mimpc`

Table 1 REACSA's physical properties

m	r	I_s	I_{RW}	$\omega_{RW,\max}$	F_n	τ_max	$t_\mathrm{on,min}$	$t_\mathrm{on,\,max}$	$t_\mathrm{off,min}$
202.81 kg	0.35 m	12.22 kg m^2	0.047 kg m^2	±250 RPM	10.36 N	1.44 N m	100 ms	300 ms	200 ms

known as dwell time constraint, can not always be brought to a full stop. Instead, a controller keeps them in a limit-cycle around the target [12].

3 Controllers

The different MPCs compared in this work are all assembled using the following generic Optimal Control Problem (OCP):

$$\underset{\substack{\mathbf{u}_0,\ldots,\mathbf{u}_{N-1},\\ \mathbf{x}_0,\ldots,\mathbf{x}_N,\\ \mathbf{e}_{u,0},\ldots\mathbf{e}_{u,N-1},\\ \mathbf{e}_{x,0},\ldots\mathbf{e}_{x,N}}}{\text{minimize}} \quad \sum_{t=0}^{N-1}\left(\mathbf{Q}^T\mathbf{e}_{x,t}+\mathbf{W}^T\mathbf{e}_{u,t}\right)+\mathbf{Q}_\mathrm{f}{}^T\mathbf{e}_{x,N} \tag{2a}$$

$$\text{subject to} \quad \mathbf{x}_{t+1}=\mathbf{x}_t+\Delta\mathrm{t}\,\mathbf{A}\mathbf{x}_t+\Delta\mathrm{t}\,\mathbf{B}(\theta_t)\,\mathbf{u}_t, \quad \forall t\in\{0,\ldots,N-1\}, \tag{2b}$$

$$-\mathbf{e}_{x,t}\le\ \mathbf{x}_t-\hat{\mathbf{x}}\le+\mathbf{e}_{x,t}, \quad \forall t\in\{0,\ldots,N\}, \tag{2c}$$

$$-\mathbf{e}_{u,t}\le\ \mathbf{u}_t\le+\mathbf{e}_{u,t}, \quad \forall t\in\{0,\ldots,N-1\}, \tag{2d}$$

$$-\tau_\mathrm{max}\le\mathbf{u}_{t,0}\le\tau_\mathrm{max}, \quad \forall t\in\{0,\ldots,N-1\}, \tag{2e}$$

$$0\le\mathbf{u}_{t,1\ldots8}\le 1, \quad \forall t\in\{0,\ldots,N-1\}, \tag{2f}$$

$$\mathbf{x}_\mathrm{lb}\le\mathbf{x}_t\le\mathbf{x}_\mathrm{ub}, \quad \forall t\in\{0,\ldots,N-1\}, \tag{2g}$$

$$\mathbf{x}_\mathrm{f,lb}\le\mathbf{x}_N\le\mathbf{x}_\mathrm{f,ub}, \tag{2h}$$

$$\mathbf{x}_0=\mathbf{x}_\mathrm{estimated}. \tag{2i}$$

where N is the prediction horizon and t the discrete-time step index. State vector $\mathbf{x}_t$ and input vector $\mathbf{u}_t$ refer to the t^th future prediction. The diagonal weight matrices $\mathbf{Q}, \mathbf{Q}_f$ and $\mathbf{W}$ weight the absolute input and state error values over the prediction horizon. The auxiliary vectors $\mathbf{e}_{x,t}$ and $\mathbf{e}_{u,t}$ are constraint by (2c) and (2d) to take the absolute value of the state error (to target state $\hat{\mathbf{x}}$) and inputs respectively. By doing so, the OCP adopts an ℓ_1-norm objective and can be cast as a Linear Program (LP), which is straightforward to solve. The forward-Euler discretized and linearized system dynamics are enforced by (2b). The RW input torque is constrained by (2e) and the thruster inputs are limited to stay between no-thrust (0) and full-thrust (1) by (2f). The flatfloor bounds, the RW's, and the system's velocity are constrained within the state-space by (2g). The recursive feasibility of the OCP is ensured by constraining the terminal state $\mathbf{x}_N$ via (2h) to be within a control invariant set as derived in [9]. The first state $\mathbf{x}_0$ is constrained to the current state estimate by (2i). The three MPCs that are derived from (2) are listed in Table 2 and explained in the following.

Table 2 Different MPCs compared in this work

Controller	Description	Solver
MIMPC	Enforces binary thruster inputs and timings explicitly	SCIP [13]
continuous MPC	Continuous MPC with ΔΣ-modulator	CLP [14]
binary informed MPC	Continuous MPC + information on ΔΣ-modulator firings in OCP	CLP [14]

The first controller is the *MIMPC* which handles the binary and timing constraints explicitly in the OCP. By doing so, the OCP (2) becomes a Mixed Integer Program (MIP) which is NP-hard to solve. Previous analyses have shown that often the optimal solution or at least a sub-optimal solution can be found for $N = 20$ within 0.1 s, hence allowing real-time control with 10 Hz [8, 9]. The details on the MIMPC can be found in [9].

The second controller is the *continuous MPC*. Here the OCP (2) stays as stated. The LP can be solved very efficiently, enabling a control loop frequency of more than 100 Hz. Consequently, the MPC outputs continuous thruster activation values and does not respect the thruster timing constraints. A $\Delta\Sigma$-modulator converts the continuous inputs into on/off values. Each thruster has its own modulator which switches the thruster state based on the integrated thruster error $u_{j,\text{error}}$ for each thruster $j = 1 \dots 8$:

$$\dot{u}_{j,\text{error}} = K \left(u_{0,j} - u_{j,\text{bin}} \right) \tag{3a}$$

$$u_{j,\text{bin}} = \begin{cases} 1, & u_{j,\text{error}} > \epsilon \\ 0, & \text{else} \end{cases} \tag{3b}$$

where $u_{0,j}$ is the continuous thruster input for the current timestep $t = 0$ determined by solving (2). The actual binary value $u_{j,\text{bin}}$ is the modulator's output. The integrator gain K is set to 1 and the trigger threshold ϵ is set to the minimum activation time $t_{\text{on, min}}$ for simplicity. Additional thruster timing logic enforces the thruster timing constraints; as soon as the modulator fires, the thrust is maintained for the minimum time. After the maximum thrust firing time, the modulator will stop firing, and after each firing, the cool-down time is enforced.

To compensate for the fact that there is now an unmodeled sub-system in the control loop, this work introduces the *binary informed MPC* as the third controller. The full $\Delta\Sigma$-modulator cannot be modeled without making (2) a MIP again. Instead, we only inform the continuous MPC about the current state of the $\Delta\Sigma$-modulator. This is achieved by simulating the behavior of the $\Delta\Sigma$-modulators forward along the prediction horizon N at each control cycle using the above logic and the current thruster errors. This results for each thruster j in the $\Delta\Sigma$-modulator-input series $u_{t,j,\text{bin}},\ t = \{0, \dots, N-1\}$. Based on this series the dynamic constraint (2b) is altered to include the planned $\Delta\Sigma$-modulator firings:

$$\mathbf{x}_{t+1} = \mathbf{x}_t + \Delta\text{t}\, \mathbf{A}\mathbf{x}_t + \Delta\text{t}\, \mathbf{B}(\theta_t)\, \mathbf{u}_t + \Delta\text{t}\, \mathbf{B}(\theta_t)\, \mathbf{u}_{t,\text{bin}}, \quad \forall t \in \{0, \dots, N-1\} \tag{4}$$

where $\mathbf{x}_t$ and $\mathbf{u}_t$ remain the decision variables of the OCP and $\mathbf{u}_{t,\text{bin}}$ is not a decision variable but the stacked vector of the forward simulated modulator behavior for all thrusters at the respective predicted timestep t:

$$\mathbf{u}_{t,\text{bin}} = \begin{bmatrix} 0 & u_{t,1,\text{bin}} & u_{t,2,\text{bin}} & u_{t,3,\text{bin}} & u_{t,4,\text{bin}} & u_{t,5,\text{bin}} & u_{t,6,\text{bin}} & u_{t,7,\text{bin}} & u_{t,8,\text{bin}} \end{bmatrix} \tag{5}$$

Overall, these three controllers represent a spectrum of how tightly binary actuation is coupled to the optimization problem. In the remainder of this work, we compare these three levels of coupling in terms of computational effort and closed-loop performance under identical experimental conditions. In all three, the horizon is set to $N = 20$, and the discretization step to $\Delta\text{t} = 0.1$ s.

4 Efficiency Analysis

To assess each controller's theoretical best performance, each controller was evaluated over 5400 simulation experiments. In each experiment, the system is initialized from the common initial state

$$\mathbf{x}_0 = [1.0,\ -0.5,\ \pi,\ 0.0,\ 0.1,\ 0,\ 0]. \tag{6}$$

The weighting matrices in (2a) are chosen as

$$\mathbf{Q} = \mathrm{diag}(1,\ 1,\ 0.12,\ \eta,\ \eta,\ 0.12\eta,\ 0)\,, \quad \mathbf{Q}_f = \xi\,\mathbf{Q}, \tag{7}$$

$$\mathbf{W} = \mathrm{diag}(0.0001,\ \kappa,\ \kappa,\ \kappa,\ \kappa,\ \kappa,\ \kappa,\ \kappa,\ \kappa)\,. \tag{8}$$

For each experiment, a single realization of η, ξ, and κ is sampled and then applied identically to all three controllers. The sampled parameters follow

$$\eta \sim \mathcal{U}(0, 0.5), \quad \xi \sim \mathcal{U}(1, 21), \quad \kappa \sim \mathcal{U}(0, 0.6). \tag{9}$$

Each controller's target is to reach the origin, i.e., $\hat{\mathbf{x}} = [0, 0, 0, 0, 0, 0, 0]$. An experiment is deemed successful when the system remains within a 0.1 m radius of the origin for a minimum of 40 s. Hereby, the value of 0.1 m is a result from [9] in which we derived the theoretical limit cycle bounds for this system, multiplied by a relaxation factor of 3. The experiment is considered to be a failure if the target has not been reached after 80 s. Upon successful completion, we report the time taken to reach the target and the average Euclidean position error of the system at the target. To determine the efficiency, the *average thruster usage* is defined as the mean duty cycle across all thrusters. This is calculated by summing each thruster's total activation time over the duration of the experiment and normalizing it by the product of the number of thrusters and the total experiment duration. The average thrust usage is determined individually for reaching the target and staying at the target. The simulated experiments are carried out under real-time conditions using the Drake toolbox [15]. In simulation, the controllers run on an x86 desktop system with an *i9-10900K* CPU with 10 cores, 3.7 GHz. As noted earlier, the LP can be solved within a few milliseconds, hence the continuous and binary informed MPCs are simulated at a control frequency of 100 Hz. In contrast, solving the MIMPC is NP-hard, so a control frequency of only 10 Hz is targeted, and the solver is terminated after 0.1 s with a possibly suboptimal solution, as described in [9].

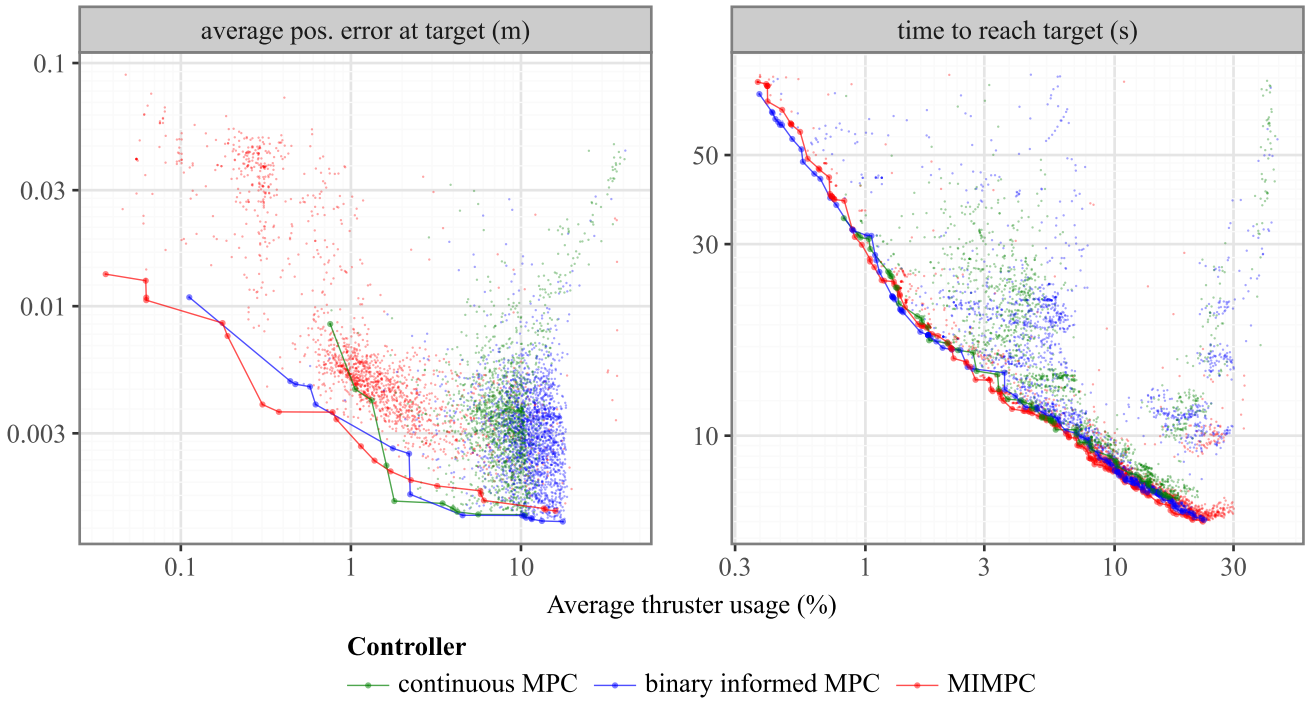

Fig. 2 Average position error after reaching and while staying at the target position (left) and time to reach the target position (right) for all compared controllers plotted over the average thrust usage. Single experiments are indicated by small dots, and the Pareto-optimal experiments for all controllers are depicted by solid lines.

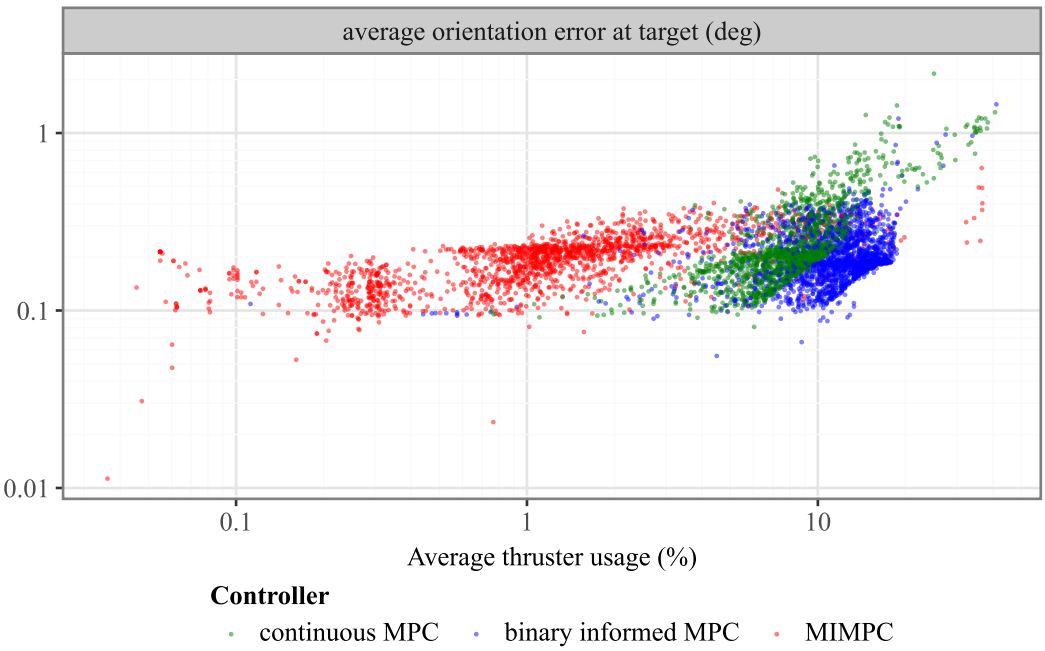

Fig. 3 Average orientation of all controllers after reaching and while doing station-keeping.

To assess the controllers' efficiencies, the time to reach the target and the position error when staying at the target are plotted along with the average thruster usages in Figure 2. The Pareto efficient experiments for each controller are highlighted and connected to a Pareto front. When comparing the *time to reach the target*, all controllers show similar efficiency as the Pareto fronts overlay. However, as indicated by the small dots representing non-Pareto-optimal experiments, the continuous MPC and the binary informed MPC scatter much higher into the inefficient region. On the low thrust side of the plot, the continuous MPC needs an average thruster usage of at least 8 % to reach the goal, which in this case, takes 34.8 s. The MIMPC and the binary informed MPC, in contrast, can use fewer thruster activations of minimum 3.7 % (MIMPC) and 3.8 % (binary informed MPC) and take 76.0 s and 70.9 s but still reach the target. On the high-thrust side, with a thrust usage of 18 %, the continuous MPC achieves its minimum time of 7 s to reach the target. With higher thrust usage and depending on tuning, the continuous MPC still reaches the goal without improving its performance. Depending on the tuning, however, it often becomes unstable and in multiple cases even exceeds the flat-floor limits. In contrast, the MIMPC and the binary informed MPC can further reduce the time to reach the goal with higher thruster usage. The average thruster activations are 22.6 % (MIMPC) and 22.8 % (binary informed MPC) with a minimum time to reach the target of only 6.1 s (MIMPC) and 6.2 s (binary informed MPC).

For the performance at the target (i.e. station-keeping), the difference in efficiencies is more distinct. The lowest average position error of 1.3 mm is reached by the binary informed MPC with an average thruster activation of 17.6 %. Again, the continuous MPC does not reach the goal with this high thruster usage and therefore only reaches a minimum error of 1.4 mm at a thruster usage of 10.5 %. Notably, comparing the MIMPC in this high thrust region of roughly >2 %, its Pareto curve lies approximately 0.1 mm to 0.3 mm above the one of continuous and binary informed MPC. This indicates a slightly lower efficiency in this region for the MIMPC. However, at a thrust usage of roughly 2 % this switches (for the binary informed MPC at a higher thrust usage than the continuous MPC, making the latter more efficient in this region). For thruster usages below 2 %, the Pareto front of the MIMPC remains below that of both binary informed and continuous MPC except for two outliers in which the Pareto fronts of MIMPC and binary informed MPC intersect. The minimum thrust usage of the continuous MPC at the target is 0.8 %, with an average position error of 8.4 mm. Both the MIMPC and the binary informed MPC reach this error, with a thrust of roughly 0.2 %, making them more efficient. Ultimately, the binary informed MIMPC reaches minimum thrust usage of 0.1 %, with an average position error of 1.1 cm, whereas the

MIMPC achieves the same error with only 0.06 % of thruster usage. The lowest thrust usage of 0.04 % at the target is achieved by the MIMPC, where it has an average position error of 1.4 cm. Notably, in the lower thrust region, the points on the Pareto fronts of the respective controllers are relatively sparse. For the continuous and the binary informed MPC, even the Pareto non-optimal experiments become very sparse in this region.

The average orientation errors at the target for all experiments are depicted in Figure 3. For an average thruster usage of <5 %, all controllers have an average error of less than 0.5°. However, since the firing of a single thruster always affects the orientation, higher thruster usage leads to an increase in orientation error, reaching up to approximately 1.8°.

5 Evaluation on the Real System

Fig. 4 Carrying out experiments at the ORGL - Images show the transition of the platform from the starting point to the origin in reading direction

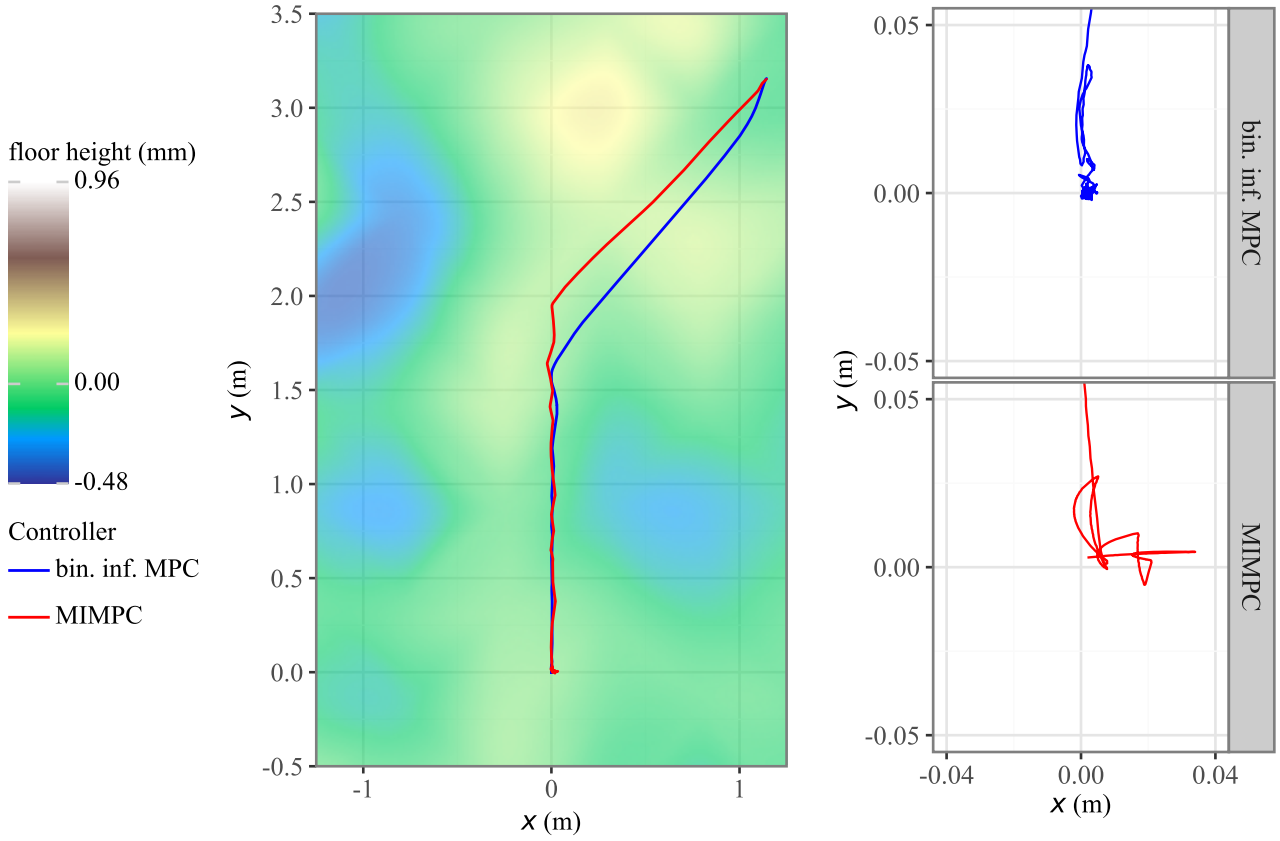

Fig. 5 Resulting trajectory of REACSA under control of the MIMPC and the binary informed MIMPC. The left plot shows the whole trajectory along with the small height variations of the otherwise flat floor, which introduce disturbances. The right plots zoom into the limit cycle at the target.

Since the binary informed MPC outperforms the purely continuous MPC in the previous analysis and the latter was unstable in certain scenarios, only the MIMPC and the binary informed MPC are evaluated and compared on the real system. In both experiments, the platform is positioned at the starting point ($x = 1.14\,\text{m}, y = 3.14\,\text{m}, \theta = 90°$). An operator then activates the air bearings manually, followed by activating the respective controller. The goal is to reach the origin at the floor's center point and de-rotate the platform to $\theta = 0°$. After REACSA has reached the origin, the controllers are kept active for another 20 s to study their limit cycle behaviors. An exemplary experiment is shown in Figure 4. The controllers have been both manually tuned with the same weights.

		MIMPC	binary informed MPC
Reaching the target	Time to reach (s)	51.268	25.209
	Total thrust usage (s)	4.90	7.51
	Average thrust usage (%)	1.2	3.7
Limit cycle at the target	Average position error (cm)	1.824	**0.853**
	Average orientation error (deg)	**0.176**	0.419
	Average thrust usage (%)	**1.4**	6.0

Table 3 Results from experiments on the real system

The resulting trajectories of REACSA under both control laws are shown in Figure 5. Table 3 contains the performance measures respectively. Videos of the experiments are available online[3]. Due to the L1-norm cost function, both controllers follow an L-shape towards the goal, as it encourages minimizing the error on all axes independently. The MIMPC follows a more rigid L-shape compared to the binary informed MPC. Overall, the binary informed MPC reaches the target of 10 cm in 25 s, which is twice as fast as the MIMPC, though it uses three times more thrust than the MIMPC.

The binary-informed MPC initially exhibits a large oscillation and then settles into a small limit cycle around the target. The resulting mean position error at the target is less than 1 cm, while the thrusters fire on average 6 % of the time. The MIMPC reaches the target and directly converges to its limit cycle. Since this cycle is larger, the resulting average position error is approximately twice that of the binary informed MPC. However, it also fires the thrusters on average for only 1.4 % of the time, which is about four times less than the binary-informed MPC. Both controllers stabilize the orientation by effectively combining the RW and the thrusters, achieving an average error below 0.5°. The MIMPC achieves an error of 0.18°, which is less than half that of the binary-informed MPC.

6 Discussion

The results highlight when it is advantageous to explicitly consider binary thrusters and timing constraints within a controller. All controllers have a similar performance in reaching the target. The continuous MPC, however, may become unstable and can even exceed constraints in high-thrust configuration. Furthermore, in low-thrust configuration, only the binary informed MPC and the MIMPC are capable of reaching the target. During station keeping at the target, in high-thrust regimes, all compared controllers achieve the most efficient experiments with similar performance. The continuous and binary informed MPC are even a bit more efficient in that region. This is probably due to the higher control frequency and the fact that the MIMPC does not always solve to optimality. The difference, however, is minimal and negligible. Since the continuous MPC in low-thrust tuning does not reach the target, only the binary informed MPC and MIMPC are successful in station keeping with low-thrust usage.

[3] https://youtu.be/briBzeDHmGs

Nevertheless, in low-thrust regimes, explicit treatment of binary inputs and thruster timings within the MIMPC is advantageous, as the results indicate that MIMPC remains the most efficient controller in this operating range. Further, the scattering of the Pareto non-optimal points indicates that in the majority of experiments, the continuous and binary informed MPC achieved high thrust and not always minimal error, despite the weights being the same as for the MIMPC. Consequently, achieving low-thrust usage at the target is more challenging with the binary informed and, especially, the continuous MPC, requiring more careful tuning compared to the MIMPC.

In summary, it can be said that the binary informed MPC is superior to the purely continuous MPC. With negligible additional computing operations, the gap in efficiency between the binary informed MPC and the MIMPC is reduced. The results on the real system show that the binary informed MPC works well in reality. It even performed better than MIMPC, although it consumed more thrust. Therefore, a definitive assessment of efficiency on the real system is not possible, requiring more experiments and more precise tuning. From a practical perspective, the choice of controller depends on mission constraints. If low computational cost or a very high accuracy is required, the binary informed MPC remains attractive as the MIMPC provides no benefit. On the other side, if robustness and fuel efficiency are important, for example, in long-duration or resource-constrained missions, then explicitly formulating the problem as a MIP is advantageous. Another point regarding computational complexity is that the continuous formulation still leaves "room" for additional complexity. For example, a quadratic cost function, which is in some sense more natural, could be advantageous and should be explored in future work. Other features, such as path following, a longer prediction horizon, or obstacle avoidance, could also be incorporated more easily. The MIMPC, on the other hand, is already close to the limit of what is computationally feasible.

7 Conclusion

This paper compared three control strategies for binary thruster systems on ESA's REACSA platform: a direct MIMPC, a continuous-input MPC with $\Delta\Sigma$-modulation, and a binary informed MPC that incorporates predicted $\Delta\Sigma$-modulator firings into the OCP. The continuous MPC performs reliably only at higher thrust utilizations and entails a risk of unstable behavior or even constraint violations. The binary-informed MPC reduces this issue by informing the OCP formulation about the modulator's effect within the prediction horizon. With substantially lower computational effort than the MIMPC, it can approach it in efficiency, while also removing the instability risk. However, in very low-thrust regimes, explicitly managing on/off decisions and timing constraints with MIMPC reduced thruster usage for a given steady-state accuracy, thereby maximized thrust efficiency. Experiments on the real system were consistent with the simulations. With identical weights, the binary informed MPC reached the goal region faster (about 25 s vs. 51 s for MIMPC) but at higher thruster usage (about 3.7 % vs. 1.2 %). In steady state, the binary informed MPC produced a smaller position error (about 0.85 cm vs. 1.82 cm), while the MIMPC yielded a smaller average orientation error (about 0.18° vs. 0.42°).

Overall, when computational resources allow, explicitly formulating the binary decision process (MIMPC) is advantageous in low-thrust and resource-constrained scenarios. If solver complexity is a concern, incorporating modulator-aware predictions into a continuous OCP is a practical alternative that recovers much of the robustness observed with MIMPC while retaining efficient LPs. Future work can explore the potential of the binary informed MPC, for instance, through more comprehensive task specifications (such as path following or obstacle avoidance) and enhanced horizon/cost designs. Additionally, the amount of information provided by the modulator to the MPC, such as when fire breaks are enforced, could be increased.

Acknowledgment

This work is partially funded by the projects: CoEx (grant number 01IW24008) funded by the German Federal Ministry for Education and Research, and ActGPT (grant number 01IW25002) funded by the Federal Ministry of Research, Technology and Space (BMFTR) and is supported with funds from the federal state of Bremen for setting up the Underactuated Robotics Lab (265/004-08-02-02-30365/2024-102966/2024-740847/2024). Further, this work has been partially supported by the German Federal Ministry of Research, Technology and Space (BMFTR) under the Robotics Institute Germany (RIG). Special thanks go to Francesca Bocconcelli, Magnus Bøgh-Larsen, and Jules Noirant for their help with the experiments at the ORGL. Many thanks also to the Robotics Section of the European Space Agency for the wonderful collaboration.

Declaration of Use of Artificial Intelligence

Artificial Intelligence was used to proofread and translate parts of this paper.

References

[1] Bong Wie. *Space vehicle dynamics and control*. Aiaa, 1998.

[2] Katsuyoshi Tsujita. A feasible study on the model predictive control for docking approach of small spacecraft using thrusters and a control moment gyro. In *2023 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, pages 696–701, June 2023. doi: 10.1109/AIM46323.2023.10196262.

[3] Avijit Banerjee, Sumeet G. Satpute, Christoforos Kanellakis, Ilias Tevetzidis, Jakub Haluska, Per Bodin, and George Nikolakopoulos. On the Design, Modeling and Experimental Verification of a Floating Satellite Platform. *IEEE Robotics and Automation Letters*, 7(2):1364–1371, Apr. 2022. ISSN: 2377-3766. doi: 10.1109/LRA.2021.3140134.

[4] Richard Zappulla, Josep Virgili-Llop, and Marcello Romano. Spacecraft Thruster Control via Sigma–Delta Modulation. *Journal of Guidance, Control, and Dynamics*, 40(11):2928–2933, Nov. 2017. ISSN: 0731-5090. doi: 10.2514/1.G002986.

[5] Anton Bredenbeck, Shubham Vyas, Martin Zwick, Dorit Borrmann, Miguel Olivares-Mendez, and Andreas Nüchter. Trajectory Optimization and Following for a Three Degrees of Freedom Overactuated Floating Platform. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4084–4091, July 2022. doi: 10.1109/IROS47612.2022.9981294.

[6] Pantelis Sopasakis, Daniele Bernardini, Hans Strauch, Samir Bennani, and Alberto Bemporad. A Hybrid Model Predictive Control Approach to Attitude Control with Minimum-Impulse-Bit Thrusters. pages 2079–2084, July 2015. doi: 10.1109/ECC.2015.7330846.

[7] Mirko Leomanni, Andrea Garulli, Antonio Giannitrapani, and Fabrizio Scortecci. All-Electric Spacecraft Precision Pointing Using Model Predictive Control. *Journal of Guidance, Control, and Dynamics*, 38(1):161–168, 2015. ISSN: 0731-5090. doi: 10.2514/1.G000347.

[8] Franek Stark, Shubham Vyas, Georg Schildbach, and Frank Kirchner. Linear Model Predictive Control for a planar free-floating platform: A comparison of binary input constraint formulations. In *Proceedings of 17th Symposium on Advanced Space Technologies in Robotics and Automation*, volume 17, Leiden, The Netherlands, Oct. 2023. doi: 10.48550/arXiv.2312.10788.

[9] Franek Stark, Shubham Vyas, Georg Schildbach, and Frank Kirchner. Mixed Integer Model Predictive Control for a free-floating platform with binary and continuous actuation. In *Proceedings of the 2024 {CEAS EuroGNC} conference*, volume 2024, Bristol, UK, June 2024. doi: 10.82124/CEAS-GNC-2024-013, `https://eurognc.ceas.org/archive/EuroGNC2024/html/CEAS-GNC-2024-013.html`.

[10] Martin Zwick, Irene Huertas, Levin Gerdes, and Guillermo Ortega. ORGL – ESA's Test Facility for Approach and Contact operations in Orbital and Planetary Environments. In *Proceedings of the International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS)*, volume 6, Madrid, Spain, June 2018.

[11] Willem Suter, Gunter Just, and Marti Vilella. REACSA: Actuated Floating Platform for Orbital Robotic Concept Testing and Control Software Development. In *Proceedings of 17th Symposium on Advanced Space Technologies in Robotics and Automation 2023*, volume 17, Scheltema, Leiden, The Netherlands, Oct. 2023.

[12] Jerry Mendel. Performance cost functions for a reaction-jet-controlled system during an on-off limit cycle. *IEEE Transactions on Automatic Control*, 13(4):362–368, Aug. 1968. ISSN: 1558-2523. Conference Name: IEEE Transactions on Automatic Control. doi: 10.1109/TAC.1968.1098940.

[13] Suresh Bolusani, Mathieu Besançon, Ksenia Bestuzheva, Antonia Chmiela, João Dionísio, Tim Donkiewicz, Jasper van Doornmalen, Leon Eifler, Mohammed Ghannam, Ambros Gleixner, Christoph Graczyk, Katrin Halbig, Ivo Hedtke, Alexander Hoen, Christopher Hojny, Rolf van der Hulst, Dominik Kamp, Thorsten Koch, Kevin Kofler, Jurgen Lentz, Julian Manns, Gioni Mexi, Erik Mühmer, Marc E. Pfetsch, Franziska Schlösser, Felipe Serrano, Yuji Shinano, Mark Turner, Stefan Vigerske, Dieter Weninger, and Lixing Xu. The SCIP Optimization Suite 9.0. ZIB-Report 24-02-29, Zuse Institute Berlin, Feb. 2024. `https://nbn-resolving.org/urn:nbn:de:0297-zib-95528`.

[14] John Forrest, Stefan Vigerske, Ted Ralphs, Lou Hafer, jpfasano, Haroldo Gambini Santos, Jan-Willem, Matthew Saltzman, a andre, Bjarni Kristjansson, h-i gassmann, Alan King, Arevall, Bohdan Mart, Pierre Bonami, Ruan Luies, Samuel Brito, and to st. coin-or/Clp: Release releases/1.17.10, Aug. 2024. doi: 10.5281/ZENODO.13347196, `https://zenodo.org/doi/10.5281/zenodo.13347196`.

[15] Russ Tedrake and Drake-Development-Team. Drake: Model-based design and verification for robotics, 2019. `https://drake.mit.edu`.