

Trajectory Optimization and Following for a Three Degrees of Freedom Overactuated Floating Platform

A. Bredenbeck^{1,2,†}, S. Vyas^{3,2}, M. Zwick², D. Borrmann¹, M.A. Olivares-Mendez⁴, A. Nüchter¹

Abstract—Space robotics applications, such as Active Space Debris Removal (ASDR), require representative testing before launch. A commonly used approach to emulate the microgravity environment in space is air-bearing based platforms on flat-floors, such as the European Space Agency’s Orbital Robotics and GNC Lab (ORGL). This work proposes a control architecture for a floating platform at the ORGL, equipped with eight solenoid-valve-based thrusters and one reaction wheel. The control architecture consists of two main components: a trajectory planner that finds optimal trajectories connecting two states and a trajectory follower that follows any physically feasible trajectory. The controller is first evaluated within an introduced simulation, achieving a 100% success rate at finding and following trajectories to the origin within a Monte-Carlo test. Individual trajectories are also successfully followed by the physical system. In this work, we showcase the ability of the controller to reject disturbances and follow a straight-line trajectory within tens of centimeters.

I. INTRODUCTION

Space debris is widely recognized as a significant challenge to all future activities in space [1]–[3]. In the extreme case, it could entirely prohibit the possibility of safe spaceflight. Becoming known as the “Kessler-Syndrome”, the described effect gained traction in recent years for multiple reasons. Firstly, today a significant portion of debris is caused by human activity. Some of the most significant contributions stem from anti-satellite missile tests and satellite collisions [4], [5]. Despite stringent requirements on de-orbiting satellites, many community members argue that these are insufficient to avoid the Kessler-Syndrome mentioned above and campaign for ASDR [6], [7].

Given the proximity of the servicing/capturing device and the respective client, the missions carry above-average risk. Hence, guaranteeing flawless operation as well as possible prior to launch is highly prioritized. For this purpose, ground-test facilities that provide system-level evaluation are necessary. The servicing/capturing system will eventually operate in a zero-g environment while all ground facilities are subject



Fig. 1. The first author at the ORGL at ESTEC. An epoxy flat-floor for floating air-bearing platforms, where slight unevenness induces disturbances.

to Earth’s gravity. Hence, precisely replicating the entire operating domain proves to be very difficult.

Air-bearing-based platforms have become the most popular testing facility that simulates microgravity, with many examples in use [8]–[13].

Figure 1 depicts one of these facilities: the ORGL at ESTEC, a European Space Agency (ESA) facility. One of the testing platforms at the ORGL provides a realistic actuator assembly, with several cold gas thrusters and a reaction wheel. This system functions as a base platform for testing new technologies or as a dummy target for capturing tests. Thus it is of interest to control the system along desired trajectories as a target or to enable the movement of the tested technology. For all air-bearing-based floating platforms, the most limiting factor for the test duration is the amount of cold gas stored onboard, which provides the necessary pressure to keep the platform floating. The compressed breathing air also functions as the propellant for the thrusters that control the position and orientation (pose) of the system. Therefore, a controller aiming to prolong the test duration must use the thrusters in a propellant-optimal manner along any trajectory. This objective is similarly vital for the control of satellites, as depletion of fuel limits the lifetime of a satellite.

The main contributions of this work are the open-source implementation and validation of a trajectory finding and following controller for free-floating systems, which are subjected to binary actuation constraints. It proposes to find optimal trajectories between two points in state-space and then use optimal control methods combined with a modulation scheme for the thrusters to follow the desired trajectory.

¹ Informatics VII, University of Würzburg, Germany

² Automation and Robotics Group, ESA, Noordwijk, Netherlands

³ Robotics Innovation Center (RIC), DFKI Bremen, Germany

⁴ SpaceR-SnT, University of Luxembourg, Luxembourg

[†] The authors acknowledge the support of Stardust Reloaded project which has received funding from the European Union’s Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 813644 and the Elite Network Bavaria (ENB) for providing funds for the academic degree program “Satellite Technology”.

TABLE I
PHYSICAL PROPERTIES OF THE SYSTEM.

Subsystem	Mass	Mol	Height	Radius
ACROBAT	154 kg	10.090 kgm ²	62.5 cm	35 cm
SATSIM	50 kg	1.416 kgm ²	20 cm	35 cm
RECAP	13.66 kg	0.67 kgm ²	20 cm	35 cm
RW	4.01 kg	0.047 kgm ²	-	-
Σ	221.67 kg	12.223 kgm ²	102.5 cm	-

The remainder of the document is structured as follows: After introducing the system model in section II, section III gives an overview of the control architecture. Section IV carries out an evaluation in simulation and on the physical system before finally summarizing and discussing the results in section V.

II. SYSTEM MODEL

The used platform is the combination of three previously existing, modular platforms: Air Cushion Robotic Platform (ACROBAT), Satellite Simulator (SATSIM), and Reaction Control Autonomy Platform (RECAP) [14]. The platforms are stacked to form the overall system, and each provides a different functionality:

- ACROBAT provides the base of the platform. Using three New Way 200 mm air-bearings which are supplied with compressed air at 4.8 bar it enables the micro-gravity behavior on the flat ground.
- SATSIM provides air tanks that supply the ACROBAT air bearing as well as the thrusters that SATSIM uses to provide thrust to the stack. SATSIM combines eight thrusters that are arranged pairwise on each side of the platform to induce forces parallel to the coordinate axes as indicated in Figure 2
- RECAP provides the Reaction-Wheel (RW) used for yaw control.

Their mass, Moment of Inertia (MoI), and size properties are shown in Table I.

This section characterizes the overall system. In particular, this section identifies the thrust by an individual thruster and the motion model.

A. Thruster Characterization

The thrusters built into this system are a combination of a tank holding pressurized gas, an intermediate pressure regulator to achieve a lower operating pressure, a regulating solenoid valve, and a Laval nozzle through which the gas escapes. Solenoid valves only allow for the states on and off. Thus, their nominal force, when opened, is of interest. At a regulated operating pressure of 5.0 ± 0.2 bar the force is measured to be 10.0 ± 0.7 N. The thrusters can hold this peak force for pulses of 100 ms but experience a significant drop for longer pulses in the order of seconds. In the following, this work assumes that the force is constant for the opening duration, and no pulses long enough to induce significant deviation from this model are considered.

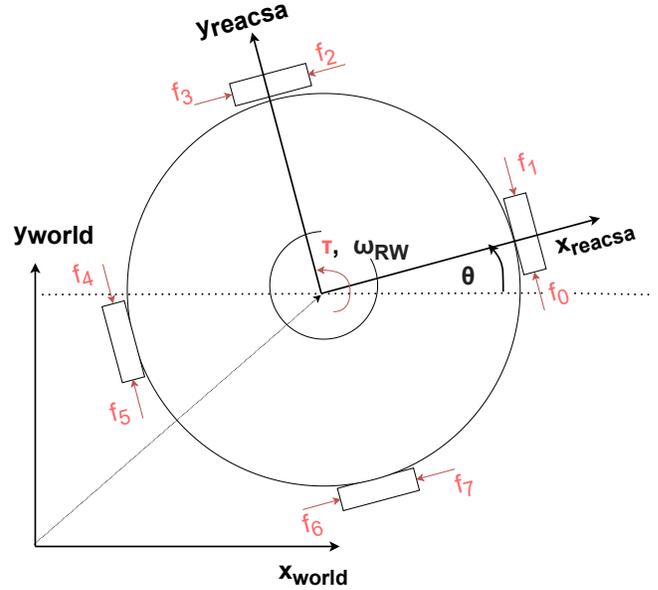


Fig. 2. Definitions of coordinate system and wrenches imposed by the actuators.

B. Dynamic Model

The following derives the overall motion model using the thrust identified above and a torque-based model for the RW. First, we define all actuators as in Figure 2. The position (x, y) of the system is defined relative to some world coordinate system, and the orientation θ is defined relative to that x axis.

Using the state and control vectors:

$$\mathbf{x} = [x \ y \ \theta \ \dot{x} \ \dot{y} \ \dot{\theta} \ \omega_{RW}]^T \quad (1)$$

$$\mathbf{u} = [\tau \ f_0 \ f_1 \ f_2 \ f_3 \ f_4 \ f_5 \ f_6 \ f_7]^T \quad (2)$$

with the forces of the thrusters f_0 to f_7 , the velocity of the reaction wheel ω_{RW} and the torque τ on the RW, the resulting continuous state equation is:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}) = \begin{bmatrix} \mathbf{0}^{3 \times 3} & \mathbf{I}^{3 \times 3} & \mathbf{0} \\ & \mathbf{0}^{4 \times 7} & \end{bmatrix} \mathbf{x} + \begin{bmatrix} \mathbf{0}^{3 \times 7} \\ \begin{matrix} 0 & \frac{-s_\theta}{m} & \frac{s_\theta}{m} & \frac{-c_\theta}{m} & \frac{c_\theta}{m} & \frac{s_\theta}{m} & \frac{-s_\theta}{m} & \frac{c_\theta}{m} & \frac{-c_\theta}{m} \\ 0 & \frac{c_\theta}{m} & \frac{-c_\theta}{m} & \frac{-s_\theta}{m} & \frac{s_\theta}{m} & \frac{-c_\theta}{m} & \frac{c_\theta}{m} & \frac{s_\theta}{m} & \frac{-s_\theta}{m} \\ \frac{-1}{I_b} & \frac{r}{I_b} & \frac{-r}{I_b} & \frac{r}{I_b} & \frac{-r}{I_b} & \frac{r}{I_b} & \frac{-r}{I_b} & \frac{r}{I_b} & \frac{-r}{I_b} \\ \frac{1}{I_w} & & & & & & & & \end{matrix} \\ \mathbf{0}^{1 \times 6} \end{bmatrix} \mathbf{u}, \quad (3)$$

where s_θ and c_θ denote the sine and cosine of the respective angle, m is the system mass, r the radius of the cylindrical body, I_w and I_b are the MoI of the RW and the overall system respectively.

III. CONTROLLER

A. Overview

The block diagram in Figure 3 gives an overview of the entire control architecture. The system consists of two main modules: the trajectory planner and the trajectory tracker. The planner computes an optimal trajectory a-priori, given system constraints specified by the user. The resulting trajectory has

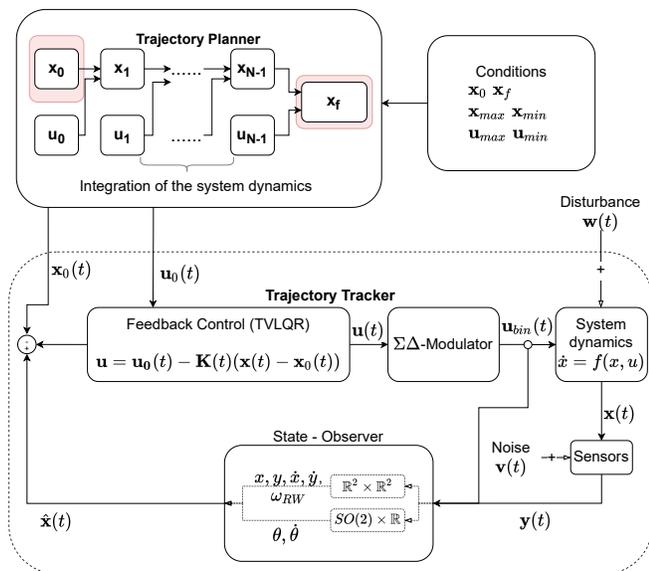


Fig. 3. Block diagram of the control architecture, consisting of a trajectory planner and follower (TVLQR, a $\Sigma\Delta$ -Modulator, and an Observer).

the form of N knot-points, each comprised of a state and a control vector, which in combination satisfy the dynamics of the system and the specified constraints if physically feasible. The trajectory tracker then tries to follow the trajectory using three sub-modules: the continuous feedback controller, the modulator, and the observer. The feedback controller computes the optimal, continuous force required to follow the trajectory, using Time-Varying Linear Quadratic Regulator (TVLQR) [15], [16]. The modulator then chooses opening times for the on/off thrusters to match the continuous force best using a $\Sigma\Delta$ -Modulation scheme [17]. A motor encoder and a Motion-Capture (MoCap) system provide measurements of the RW velocity and the system pose, respectively. The observer uses the most recent measurements available at the sensors and the commanded control input to optimally estimate the current system state. The state estimates and the torque commands are computed at 100 Hz while the force command is calculated at a slower 10 Hz to better abide by the physical limitations of the thrusters. Further, the trajectory follower can follow the previously computed optimal trajectory and any other admissible (physically feasible) trajectory.

B. Trajectory Planner: Optimizing over Discrete States and Control Inputs

1) *Cost Function*: The cost function J is the main factor determining the shape of the resulting trajectory. The two qualitative criteria that should be optimized are propellant efficiency and trajectory duration. Given the actuator limits, there is a minimal time for the system to reach the final state. This trajectory is called time-optimal. On the other hand, if the desired finishing time is infinite, the most propellant-efficient action is to do nothing. In order to find a trajectory that satisfies both criteria, the approach is as follows:

First, find the time-optimal trajectory using the cost function $J = t_f$. In addition, it is enforced that $t_f \geq 0$. It is well known that the time-optimal solution for such systems is a “bang-bang” controller [18] – a controller jumping between maximal and minimal control values. However, it also provides a lower bound for the overall time required to follow the trajectory.

Afterward, to define a desired final time, the time-optimal final time is multiplied with a buffer factor α : $t_{f,des} = \alpha \cdot t_f^*$, where the buffer factor α is chosen heuristically to provide slow, smooth movement. The number of tunable heuristics is the main difference between this approach and simultaneously optimizing the final time and actuation costs. Simultaneous optimization requires two separate weights for both cost functions, whereas the separation reduces this to one.

The goal is to find the trajectory that minimizes the propellant usage, finishing at the desired time. The force exerted by the thrusters is approximately proportional to the propellant used; therefore, a reasonable proxy for minimal-propellant is minimal force. The cost function is then

$$J = \sum_{k=1}^N \mathbf{u}_k \mathbf{R} \mathbf{u}_k^T, \quad (4)$$

where \mathbf{R} is a diagonal matrix that contains the respective weights for each control value. By choosing a large weight for all thrusters and a small weight for the RW the actuation of the thrusters is minimized.

2) *Direct Collocation*: To find an optimal trajectory that minimizes the above cost functions, we use direct collocation [19], [20]. Its key aspect is to discretize the trajectory at N instances of time t_k , which are denoted as knot-points. Each knot-point is subject to constraints that reflect the maximal and minimal state and control values and constraints concerning its neighbors. In particular, the dynamic model (as derived in section II) must be satisfied. Since that model is continuous, we use Hermite-Simpson collocation [19] to integrate its dynamics. Then it is ensured that the system dynamics are satisfied for any set of subsequent knot-points. In other words, the state and the control applied at some knot-point must result in the state at the next knot-point according to the integrated dynamics. From this it is possible to derive a minimization problem on all $N \times (\text{Dim}(\mathbf{x}) + \text{Dim}(\mathbf{u}))$ variables. For the time-optimal problem, the final time adds one more decision variable. However, the optimization problem that incorporates the binary restriction on the actuators is of the class Mixed-Integer Non-Linear Programming (MINLP). These problems are increasingly difficult to solve because they are NP-hard and combine challenges of handling non-linearities with a combinatorial explosion of integer variables [21]. By relaxing the binary condition on the respective control variables and assuming them to be continuous, the problem reduces to a problem of the class Non-Linear Programming (NLP), which can be solved significantly faster. The resulting optimization problem over all states X and control values U at all knot-

points is:

$$\begin{aligned}
& \min_{X,U} \{J(X,U)\} \quad \forall k \in [0, N-1] \text{ s.t.} \\
& \mathbf{x}(0) = \mathbf{x}_{init}, \quad \mathbf{x}(t_f) = \mathbf{x}_{final} \\
& \mathbf{x}_{min} \leq \mathbf{x}_k \leq \mathbf{x}_{max}, \quad \mathbf{u}_{min} \leq \mathbf{u}_k \leq \mathbf{u}_{max} \\
& \mathbf{x}_{k+1} - \mathbf{x}_k = \frac{\Delta t}{6} (\mathbf{f}_k + 4\mathbf{f}_{k+1/2} + \mathbf{f}_{k+1}) \\
& \text{where } \mathbf{x}_{k+1/2} = \frac{1}{2} (\mathbf{x}_k + \mathbf{x}_{k+1}) + \frac{\Delta t}{8} (\mathbf{f}_k - \mathbf{f}_{k+1}) \\
& \text{and } \mathbf{u}_{k+1/2} = \frac{1}{2} (\mathbf{u}_k + \mathbf{u}_{k+1})
\end{aligned} \tag{5}$$

Readily available, open-source solvers, such as IPOPT (Interior Point OPTimiser) [22], using the programming interface provided by Drake [23], find a solution for a zero initial- and final velocity trajectory within ten seconds. However, this relaxation yields trajectories that demand continuous control input which the discrete or binary actuator cannot provide. Therefore, any trajectory tracking controller needs to consider how to translate the desired control input into the discrete space. This is further discussed in subsection III-C.

C. Trajectory Follower

1) *Time Varying Feedback Controller*: Once there is an optimal trajectory to be followed, the robot should find the controls that, given the current state, propels the robot along the trajectory. We propose to use a full state feedback controller based on the TVLQR formulation. Solving the Differential Riccati Equation (DRE) by initializing the cost-to-go with its value at the final time and integrating it backward in time yields the respective cost-to-go matrices at different times t and from this, one can compute the feedback gain matrices $\mathbf{K}(t)$ [16]. The resulting control law is:

$$\mathbf{u}(t) = \mathbf{u}_0(t) + \mathbf{K}(t)(\mathbf{x}(t) - \mathbf{x}_0(t)), \tag{6}$$

where $\mathbf{u}(t)$ is the desired control input, $\mathbf{u}_0(t)$ is the feed-forward control from the pre-computed trajectory, $\mathbf{x}(t)$ is the current state, and $\mathbf{x}_0(t)$ is the desired state according to the trajectory. Note that solving the DRE backward in time implies the controller must pre-compute the gain matrices for a trajectory beforehand, resulting in a few additional seconds of computation time during the initialization.

2) *Sigma-Delta-Modulator*: Given that there are discrete actuators in the present system, one needs to answer how to modulate the continuous control signal derived in the previous section. One approach introduced in [17] is using a $\Sigma\Delta$ -Modulator, a technique commonly used in analog to digital modulation, to modulate the continuous force onto the binary actuators. The basic concept of a $\Sigma\Delta$ -Modulator is to trigger a pulse as soon as the integrated error between the desired and current output reaches some specific threshold. The general concept of the $\Sigma\Delta$ -Modulator is the following:

- 1) Sample the continuous signal $u(t)$ at some frequency f_{smp} using sample and hold.
- 2) Compute the error $e_{\Sigma\Delta}(t)$ by taking the difference of the current thruster output $y_{Thruster}(t)$ (which is already modulated) to the desired value.

- 3) Integrate the error by numeric integration where Δt is the time difference between two samples:
 $w_e(t) = \int_{t_0}^t e_{\Sigma\Delta}(\tau) d\tau \approx \Delta t \sum_{i=0}^T e_{\Sigma\Delta}(i \cdot \Delta t)$.
- 4) Once the integrator value surpasses some threshold, i.e. $w_e(t) > \epsilon$, trigger a pulse. We guarantee equivalence of the integrals of the continuous and modulated signal by choosing the threshold such that a single pulse resets the error integrator to zero. Thus the threshold must be the inverse of the control frequency times the nominal force, i.e., $\epsilon = \frac{f_{nom}}{f_{smp}}$.

Finally, the modulator feeds back the current output value, closing the loop.

3) *State-Estimation*: This work assumes that the entire state is available for feedback in the previous sections. This assumption is not generally satisfied since the MoCap system only provides pose estimates (x, y, θ) , and numerically differentiating those is not sufficient [24]. Further, all available measurements are subject to noise and thus require some filtering process to improve their quality given some underlying system model. The standard for this filter is the Kalman Filter (KF) [25]. By combining the knowledge of the underlying system and the measurements optimally, in the sense of a quadratic estimation error, the filter smoothes the data and rejects extreme outliers that fall significantly outside the distribution of the specified measurement error. This work combines a classical KF for the position, its derivative, and the RW velocity with a KF over the Lie Algebra $SO(2) \times \mathbb{R}$ for the orientation and the angular velocity as given in [26] to achieve full state estimation.

IV. RESULTS

A. Simulation

Before being evaluated on the physical system, this work evaluates the controller in a Gazebo [27] simulation that incorporates sensor noise and the slight unevenness of the floor. The simulated sensors are subjected to Additive White Gaussian Noise (AWGN) with variances that match the measured variances of the real system ($\sigma_x^2 = \sigma_y^2 = 1 \times 10^{-5} \text{ m}^2$, $\sigma_\theta^2 = 1 \times 10^{-5} \text{ rad}^2$ and $\sigma_{\omega_{RW}}^2 = 1 \times 10^{-4} (\text{rad/s})^2$).

The simulation includes a heightmap representing the last openly available measurements from [28] to incorporate the unevenness of the flat-floor, which has a maximal deviation of 1 mm over one meter. For more details about the simulation, the interested reader is referred to [29].

1) *Stabilization*: The system is simulated while being subjected to the instantaneous disturbance consisting of a force and a torque $\mathbf{d} = [\mathbf{f}_d \quad \tau_d]$ and lasting for Δt , where:

$$\mathbf{f}_d = [5000 \text{ N} \quad 5000 \text{ N}]^T, \quad \tau_d = 1000 \text{ N m}, \quad \Delta t = 0.001 \text{ s.} \tag{7}$$

The response is shown in Figure 4 and the actuation of the controller in Figure 5. Accompanying each plot of the controlled system response is a plot of the system response to the disturbance without being controlled to highlight the effect of the controller.

As Figure 4 shows, the controller succeeds in bringing the system back to the origin after the initial disturbance. It

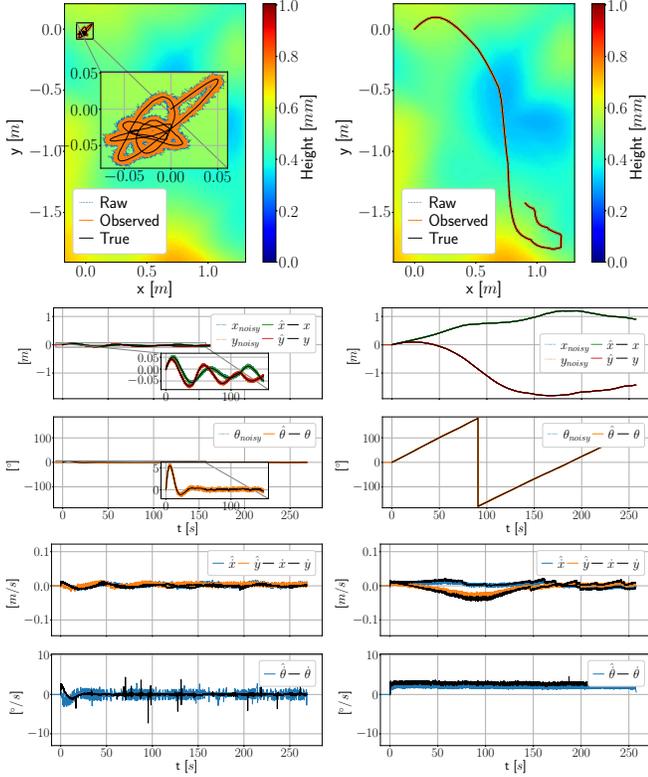


Fig. 4. Ground-track, individual coordinates, and velocities of the system responding to a disturbance (at $T = 0$ s) on an uneven floor in simulation. Left: Controller stabilizing the system at the origin. Right: No controller running. An animation of the stabilization process is given at <https://youtu.be/KRYcq3VjQUo?t=4>.

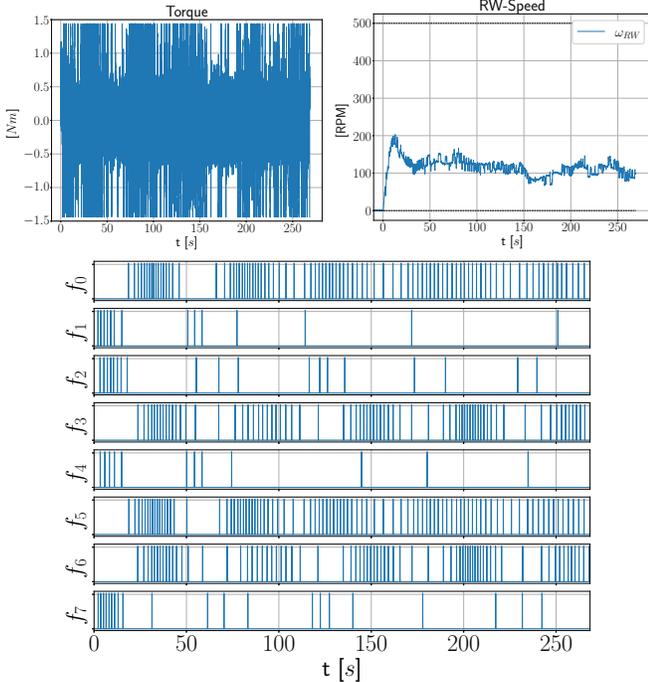


Fig. 5. Actuation required to stabilize the system at the origin on an uneven floor in simulation while being subjected to a disturbance (cf. Figure 4). Top Left: the torque exerted by the motor onto the RW. Top Right: the resulting RW velocity (raw and observed). Bottom: the thruster activity for all eight thrusters.

maintains the system within less than 10 cm and 6° of the origin. The unactuated system, on the other hand, drifts over 1.5 m and performs more than a full rotation. The offset from the origin of the controlled system is attributed to two factors: the thruster’s binary nature and the unevenness of the floor. The continuous actuation needs to accumulate in the modulator until a firing is triggered, propelling the system through the origin, and the process repeats in the opposite direction. This well-known behavior for thruster-based systems is referred to as a limit cycle [30]. At the same time, the controller needs to compensate for the disturbances induced by the flat-floor, further adding to the oscillations about the origin. In Figure 5 the torque clearly shows “bang-bang” behavior. Since the inertia of the RWs is two orders of magnitude smaller than the system’s inertia, the controller demands high torques quickly alternating between maximal and minimal torque. However, the RW velocity remains within its allowable bounds. Also, in Figure 5, the thrusters 1, 2, 4, and 7 initially show high activity since they are the ones pointed in negative x and y direction, thus counteracting the disturbance. Afterward, the thrusters show alternating behavior corresponding to the small oscillations about the origin. Finally, the thrusters regularly fire to move the system towards the positive y direction (Thrusters 0 and 5) and slightly less frequently towards the positive x direction (Thrusters 3 and 6), which corresponds to the gradient of the floor at the origin.

2) *Monte Carlo*: To demonstrate the generalizability of the controller to different initial locations on the flat-floor the system also undergoes a Monte Carlo test simulation. For a large ($n = 100$) number of episodes, the robot spawns at random initial poses, whereby x, y , and θ result from drawing from uniform random distributions that span the range $[-2, 2]$, $[-4, 4]$, and $[-\pi, \pi]$ respectively. The controller computes the optimal trajectory to the origin and starts following it. An episode is successful when the euclidean distance to the origin, the euclidean velocity as well as the angular error and velocity are smaller than the threshold ϵ :

$$\begin{aligned} & [\epsilon_{lin} \quad \epsilon_{ang} \quad \epsilon_{lin,vel} \quad \epsilon_{ang,vel}] \\ & = [0.05 \text{ m} \quad 0.05 \text{ m s}^{-1} \quad 0.05 \text{ rad} \quad 0.05 \text{ rad s}^{-1}] \quad (8) \end{aligned}$$

Figure 6 shows the results of the Monte Carlo test. During the Monte Carlo simulation, the controller commands the system to the origin from all 100 tested initial conditions. As seen in Figure 6 the trajectories are mostly straight lines with some minor deviations caused by the uneven ground, combined with the noise of the system and the binary thrusters. One particular trajectory (purple, bottom right of the ground-track) overshoots a desired state along the trajectory at approximately 38 s. Since the controller only attempts to control the system to the desired state at some time (single planning approach), it returns to a previous location resulting in a loop. This loop also corresponds to the spikes in velocities (purple). All trajectories reach the desired region at the origin and slow down to the desired maximal velocity in less than 140 s.

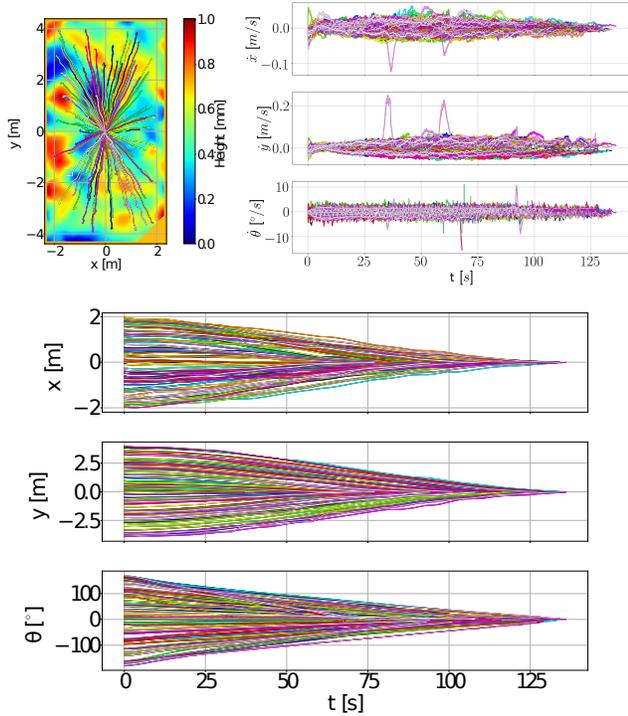


Fig. 6. Monte Carlo simulation of following trajectories to the origin. Each optimal trajectory is computed using $N = 100$ knot-points.

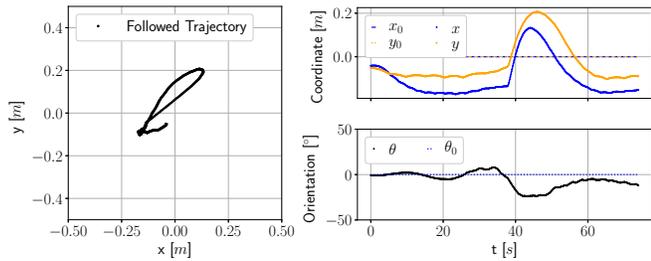


Fig. 7. Ground-track and individual coordinates of the controller stabilizing the physical system at the origin. On the right the desired value is indicated as a dashed line. A video is given at <https://youtu.be/KRYcq3VjQUo?t=168>.

B. Physical System

1) *Experimental Setup:* The RW experiences stiction at low revolutions per minute. Thus, to avoid issues, the RW is spun up to half its rated velocity before each trajectory. From this state, the experimenter places the system manually at the origin of the coordinate system before starting the trajectory follower.

2) Results:

a) *Stabilization:* This first section also evaluates the system's stabilization capabilities for the physical system. The disturbance is added by manually pushing the system. The results are shown in Figures 7 and 8.

The system generally exhibits the tendency to drift in negative x and y directions in all experiments. This tendency implies a slope pointing in this direction, which the heightmap from [28] further supports. During the stabilization process, the system initially stabilizes within 15 cm and 15° of the

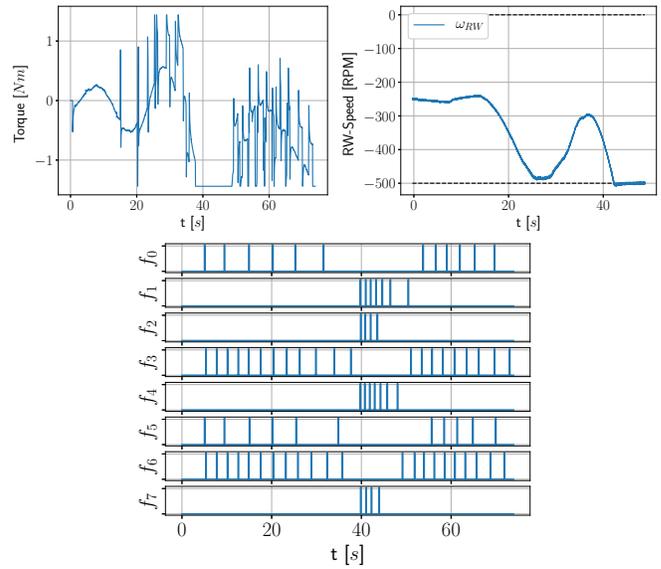


Fig. 8. Actuation of stabilizing the physical system (cf. Figure 7).

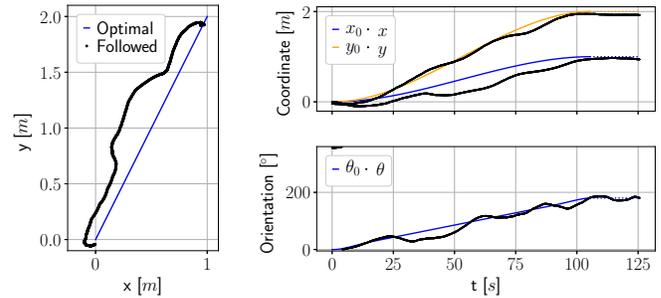


Fig. 9. Ground-track and individual coordinates of the controller following a straight-line trajectory on the physical system. After reaching the final pose of the trajectory the desired value is indicated as a dashed line. A video is given at <https://youtu.be/KRYcq3VjQUo?t=193>.

origin before the disturbance. The disturbance moves the system about 35 cm, and within 30 s the system stabilizes in the initial region again. Figure 8 shows that very little thrusting is required to achieve this result, firing no thruster more than once every two seconds during the entire process. The RW, on the other hand, saturates rather quickly after the disturbance. It quickly reaches its maximum rotational velocity as it attempts to absorb the entire angular momentum put into the system by the disturbance. Afterward, the thrusters perform the attitude control themselves; thus, the pointing accuracy degrades as larger errors are required to trigger a pulse since thruster usage is penalized more than RW acceleration.

b) *Straight-Line Trajectory:* Further, this section evaluates the physical system's performance following a straight-line trajectory (similar to the ones from section IV-A.2). The results are shown in Figures 9 and 10.

The average Euclidean and angular error of the physical system to the desired trajectory are 0.325 m and 23.8° respectively, in a trajectory covering 2.2 m and 180° . In particular, the error in the x -axis is significant. This error

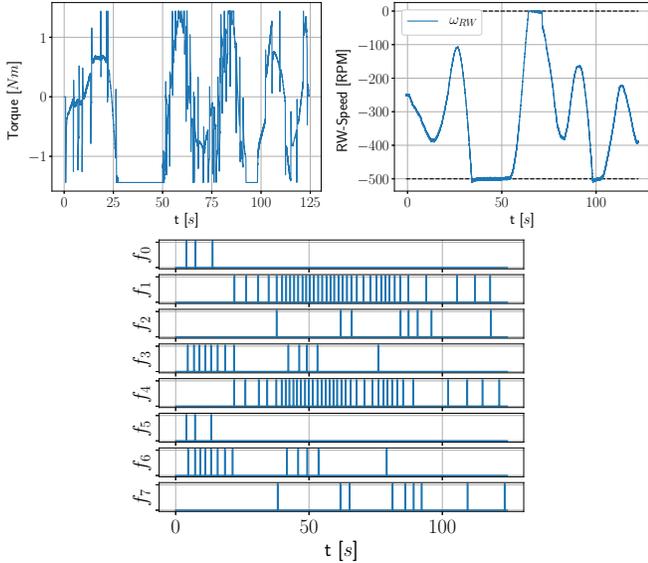


Fig. 10. Actuation of following the straight-line trajectory (cf. Figure 9) on the physical system.

TABLE II
OPTIMAL AND REAL THRUSTER ON-TIME (CF. FIGURE 9).

Thruster	0	1	2	3	4	5	6	7	Σ
Optimal	0.499 s	0.004 s	0.061 s	0.296 s	0.004 s	0.499 s	0.296 s	0.061 s	1.72 s
Real	0.30 s	3.40 s	0.70 s	1.30 s	3.40 s	0.30 s	1.30 s	0.70 s	11.40 s

is due to the slope in that region of the flat-floor, with a gradient in the negative x -direction. As in the stabilization case, the heightmap shows a local minimum adjacent to the trajectory.

Further, the RW saturates at both ends of the allowable range throughout the trajectory. At each instance, the orientation deviates more significantly from the desired orientation, implying that large desired changes in angular momentum of the entire system quickly lead to RW saturation. The controller then performs orientation control using only thrusters, thus, losing precision. However, at the end of the trajectory, the RW recovers from saturation, and the pointing accuracy at the final state is within 10° . The thrusters were firing appropriately, never exceeding one fire per second for an individual thruster. Moreover, when considering the thruster on-times, as depicted in Table II, it shows an increase of less than one order of magnitude for the overall thruster on-times compared to the optimal value. Some individual thrusters (0, 5) even decrease on-time, whereas others increase by one or multiple orders of magnitude while compensating for uneven ground. Note that the long on-times for thrusters 1 and 4 stem from compensating the previously mentioned error in x -direction since these thrusters are enacting a force in positive x -direction throughout the middle section of the trajectory.

c) Discussion: The experiments demonstrate that the general behavior of the simulation and the physical system is similar, indicating that the simulation is a realistic representation of the scenario. The controller manages to stabilize the system and follow a pre-computed trajectory. However, the deviations from the set values are more significant in

the physical system. Besides shortcomings in the simulation, such as the idealized thrusters, the limited heightmap resolution, and a single ground-contact point, other factors that influence this difference stem from the hardware. The most significant factors that influence the overall system performance are the lack of precise system identification and limited control authority. The former is outside the scope of this work; thus, it relies on previous inertial measurements of the system. An offset here contributes to the size of the limit-cycle the system exhibits around the desired orientation. This effect is particularly powerful since the control architecture combines the controller with a KF, such that the model errors accumulate and deteriorate performance [31]. Improving this model regarding inertial parameters, thrust, and thrust vectors for individual thrusters will significantly improve the system performance.

The latter is a consequence of the initial design of the floating platform. The system's high mass means that the thrusters can hardly compensate for the effects of any unevenness of the floor. For example, on a slope of $1 \frac{\text{mm}}{\text{m}}$ the system experiences a constant disturbance force of approximately 2.2 N. In an ideal scenario aligning two thrusters exactly with this disturbance force, they can provide approximately 20 N in the opposing direction. Thus to compensate for the disturbance force, the thrusters would have to have an on-time of at least 10%. This firing rate is already larger than the firing rate observed for the followed straight-line trajectory.

Additionally the RW saturates very fast. Given the inertias of the entire system and the RW it can compensate only for a slight change in angular velocity of approximately 6°s^{-1} . Despite the trajectory planner incorporating this limitation, any disturbance that imposes an equivalent torque on the system quickly saturates the RW and thus degrades the control authority of the system. One source of such disturbance is the uneven floor. Another is an unequal nominal force of individual thrusters. When firing two thrusters that point the same way (e.g., thrusters 0 and 5), the exerted wrench should only contain a force pointing in one direction. However, assuming a difference between the two thrusters, it also contains a disturbance torque τ_d . If this difference is only 10% and one assumes the 10% thruster on-time to compensate for the floor unevenness, this will saturate the RW within 34 s of operation.

All in all, the evaluation shows that this control architecture, while limited by immaturity aspects, is effective for a free-floating system with binary actuation constraints.

V. CONCLUSION & SUMMARY

Three Degrees of Freedom (DoF) floating platforms are a good way of partially emulating microgravity environments as they are present in space applications. This work introduces a controller for one of the heaviest floating platforms in Europe located within the ORGL at ESTEC, ESA.

First, this work proposes a dynamic model of the overall system, actuated by eight solenoid-valve thrusters and one RW. Said controller consists of two main components: the trajectory planner and the trajectory follower. The former

pre-computes optimal trajectories that connect two arbitrary states while minimizing the force exerted by the thrusters. The trajectory follower computes continuous control actions to follow these and any other physically feasible trajectory using a TVLQR formulation. To abide by the binary nature of the thrusters, it uses $\Sigma\Delta$ -modulation to transfer the continuous force command onto the thrusters. Finally, a KF estimates the system state, providing feedback for the control architecture.

This work further develops a simulation of the overall system that takes measurement noise and the unevenness of the floor into account. When testing the control architecture in this simulation, the controller achieves at least 16 cm average Euclidean and 5° average angular error. The controller proves to be robust to arbitrary initial poses on the flat-floor. In a Monte-Carlo simulation where the robot is spawned at arbitrary initial poses and tasked with finding and following optimal trajectories to the origin, the controller achieves a 100% success rate.

The controller is further evaluated on the physical system. A straight-line trajectory is also followed successfully but experiences a drop in performance compared to the simulation. The average euclidean error approximately doubles, and the angular error increases by one order of magnitude. The increase in error is mostly attributed to a lack of precise system identification and control authority.

A thorough system identification, including a more precise characterization of individual thrusters and their thrust vectors, will improve the system's performance in the future. This improved model is helpful in two domains: the output trajectories of the planner will be more representative of the physical system, and the controller will follow these better. Redesigning some system components to increase control authority will also help to improve accuracy. The two most promising options are decreasing the overall weight and inertia of the system and increasing the reaction wheel inertia.

Further work at the ORGL intends to implement, test, and compare multiple implementations based on the software framework developed in this work. More specifically, the future work will include improving the state estimation by adding additional sensors such as a Inertial Measurement Unit (IMU) and eventually decoupling it from the global positioning system. One approach to improve the control will also be to solve the trajectory optimization problem online and control the system in a Model Predictive Control (MPC) fashion, increasing resilience to disturbances.

REFERENCES

- [1] R. Crowther, "Space junk—protecting space for future generations," *Science*, vol. 296, no. 5571, pp. 1241–1242, 2002.
- [2] C. J. Newman and M. Williamson, "Space sustainability: Reframing the debate," *Space Policy*, vol. 46, pp. 30–37, 2018.
- [3] D. J. Kessler and B. G. Cour-Palais, "Collision frequency of artificial satellites: The creation of a debris belt," *Journal of Geophysical Research: Space Physics*, vol. 83, no. A6, pp. 2637–2646, 1978.
- [4] G. Neuneck, *China's ASAT test — A warning shot or the beginning of an arms race in space?* 2008.
- [5] T. Kelso, "Analysis of the iridium 33 cosmos 2251 collision," 2009.
- [6] J. Chatterjee, J. N. Pelton, and F. Allahdadi, *Active Orbital Debris Removal: Active orbital debris removal and the Sustainability of Space*, pp. 921–940. Cham: Springer International Publishing, 2015.
- [7] C. P. Mark and S. Kamath, "Review of active space debris removal methods," *Space Policy*, vol. 47, pp. 194–206, 2019.
- [8] K. Yoshida, "ETS-VII flight experiments for space robot dynamics and control," *Experimental Robotics VII*, pp. 209–218, 2001.
- [9] "Air bearing floor." https://www.nasa.gov/centers/johnson/engineering/integrated_environments/air_bearing_floor/index.html. Accessed: 2021-08-18.
- [10] E. Papadopoulos, I. Paraskevas, T. Flessa, K. Nanos, Y. Rekleitis, and I. Koutolatis, "The NTUA space robot simulator: Design & results," *Proceedings of the International Conference on Intelligent Robots and Systems (IROS)*, 2011.
- [11] A. Redah, H. R. Ramavaram, and S. Montenegro, "Robotic testing platform for autonomous rendezvous and docking of floating vehicles," *Proceedings of the i-SAIRAS*, 2019.
- [12] S. Wehrmann and M. Schlotterer, "Coordinated orbit and attitude control of a satellite formation in a satellite simulator testbed," *Proceedings of the 10th International ESA Conference on Guidance, Navigation & Control Systems*, 05 2017.
- [13] T. Rybus and K. Seweryn, "Planar air-bearing microgravity simulators: Review of applications, existing solutions and design parameters," *Acta Astronautica*, vol. 120, pp. 239–259, 2016.
- [14] M. Zwick, I. Huertas, L. Gerdes, and G. Ortega, "ORGL - ESA's test facility for approach and contact operations in orbital and planetary environments," *Proceedings of the i-SAIRAS*, 2018.
- [15] D. Bertsekas, *Dynamic Programming and Optimal Control: Approximate dynamic programming. Volume 2*. Athena Scientific, 2012.
- [16] R. Tedrake, "Underactuated robotics: Algorithms for walking, running, swimming, flying, and manipulation (course notes for mit 6.832)." Downloaded on 21.12.2021 from <http://underactuated.mit.edu/>.
- [17] R. Zappulla, *Experimental Evaluation Methodology for Spacecraft Proximity Maneuvers in a Dynamic Environment*. PhD thesis, Naval Postgraduate School Monterey United States, 2017.
- [18] J. Betts, *Practical Methods for Optimal Control Using Nonlinear Programming, Third Edition*. Advances in Design and Control, Society for Industrial and Applied Mathematics, 2020.
- [19] C. Hargraves and S. Paris, "Direct trajectory optimization using nonlinear programming and collocation," *AIAA J. Guidance*, 1987.
- [20] M. Kelly, "An introduction to trajectory optimization: How to do your own direct collocation," *SIAM Review*, vol. 59, pp. 849–904, 01 2017.
- [21] P. Belotti, C. Kirches, S. Leyffer, J. Linderoth, J. Luedtke, and A. Mahajan, "Mixed-integer nonlinear optimization," *Acta Numerica*, 2013.
- [22] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical Programming*, vol. 106, pp. 25–57, 2006.
- [23] R. Tedrake and the Drake Development Team, "Drake: Model-based design and verification for robotics," 2019.
- [24] F. Van Breugel, J. N. Kutz, and B. W. Brunton, "Numerical differentiation of noisy data: A unifying multi-objective optimization framework," *IEEE Access*, vol. 8, pp. 196865–196877, 2020.
- [25] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Transactions of the ASME—Journal of Basic Engineering*, vol. 82, no. Series D, pp. 35–45, 1960.
- [26] I. Markovic, J. Cacic, and I. Petrovic, "On wrapping the kalman filter and estimating with the SO(2) group," *CoRR*, vol. abs/1708.05551, 2017.
- [27] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," 2004.
- [28] H. Kolvenbach and K. Wormnes, "Recent developments on ORBIT, a 3-dof free floating contact dynamics testbed," *Proceedings of the i-SAIRAS*, 2016.
- [29] A. Bredenbeck, S. Vyas, M. Zwick, D. Borrmann, M. Olivares-Mendez, and A. Nüchter, "Finding and following optimal trajectories for an overactuated floating robotic platform," in *16th Symposium on Advanced Space Technologies in Robotics and Automation*, 2022.
- [30] S. W. Jeon and S. Jung, "Hardware-in-the-loop simulation for the reaction control system using pwm-based limit cycle analysis," *IEEE Transactions on Control Systems Technology*, 2012.
- [31] J. Doyle, "Guaranteed margins for LQG regulators," *IEEE Transactions on Automatic Control*, vol. 23, no. 4, pp. 756–757, 1978.