

Robust Adaptive Predictive Control for Hook-Based Aerial Transportation Between Moving Platforms

Péter Antal , Andrea Carron , Melanie Zeilinger , *Member, IEEE*, Roland Tóth , *Fellow, IEEE*, and Tamás Péni , *Member, IEEE*

Abstract—This letter presents a novel model predictive control (MPC) approach for autonomous pick-and-place between moving platforms with a hook-equipped aerial manipulator. First, for accurate and rapid modeling of the complex dynamics, a digital twin model of the quadcopter equipped with a hook-based gripper, implemented in MuJoCo, is constructed and used as the predictive model for the MPC. To handle uncertainties of the predictive model (e.g. due to aerodynamics and uncertain payloads), a robust adaptive MPC approach is proposed. By systematic integration of zero-order robust optimization (zoRO) based uncertainty propagation and an extended Kalman filter (EKF) for parameter estimation, the MPC algorithm ensures robust constraint satisfaction, high performance, and computational efficiency. The effectiveness of the proposed method is evaluated in complex simulated scenarios and in real-world flight experiments.

Index Terms—Aerial systems: mechanics and control, optimization and optimal control, planning under uncertainty.

I. INTRODUCTION

UNMANNED aerial vehicles (UAVs), and in particular quadrotors, are increasingly applied to solve complex tasks that require robotic manipulation and physical interaction with the environment, such as fixing high-voltage electric lines, cleaning windows, repairing blades of wind turbines, or package delivery [1]. In this work, we consider the challenging problem of dynamic pick-and-place between two moving ground platforms using a quadrotor equipped with a hook-based manipulator, as it is illustrated in Fig. 1.

Received 22 April 2026; accepted 15 August 2026. Date of publication 2 September 2026; date of current version 15 September 2026. This article was recommended for publication by Associate Editor C. Ioan Vasile and Editor L. Pallottino upon evaluation of the reviewers' comments. This work was supported by NCCR Automation, under Grant Agreement 51NF40_225155 from Swiss National Science Foundation and in part by the Ministry of Science and Technology of Hungary from the National Research, Development and Innovation Fund under Grant MEC_R 155483. (*Corresponding author: Péter Antal.*)

Péter Antal and Tamás Péni are with the Systems and Control Lab, HUN-REN Institute for Computer Science and Control, 1111 Budapest, Hungary (e-mail: antalpeter@sztaki.hu; peni@sztaki.hu).

Andrea Carron and Melanie Zeilinger are with the Institute for Dynamic Systems and Control, ETH Zürich 8092 Zürich, Switzerland (e-mail: carrona@ethz.ch; mzeilinger@ethz.ch).

Roland Tóth is with the Systems and Control Lab, HUN-REN Institute for Computer Science and Control, 1111 Budapest, Hungary, and also with the Control Systems Group of the Eindhoven University of Technology, 5612 AP Eindhoven, The Netherlands (e-mail: r.toth@tue.nl).

Video: <https://youtu.be/KuwKfKcUoDY> Code: <https://github.com/AIMotionLab-SZTAKI/rampc-aerial-pick-and-place>

This article has supplementary downloadable material available at <https://doi.org/10.1109/LRA.2026.3730216>, provided by the authors.

Digital Object Identifier 10.1109/LRA.2026.3730216

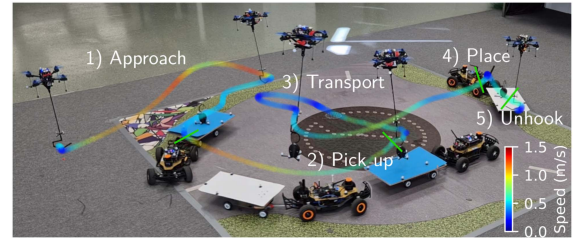


Fig. 1. Dynamic pick-and-place with a hook-based aerial manipulator.

Aerial pick-and-place can be decomposed into three tasks: pick-up, transportation, and placement. To complete these tasks, two main configurations have become dominant: unmanned aerial manipulators (UAMs) by attaching robotic extensions to the quadcopter body [2], [3], [4], [5], and cable-suspended transportation systems [6], [7], [8], [9]. The full actuation and versatility of UAMs facilitate reliable and autonomous pick-and-place [4], [5], but their relatively high weight and power consumption limit their flight time and maneuverability.

Cable-suspended systems, by contrast, have recently demonstrated remarkable agility and precision. Early methods relied on geometric control, exploiting differential flatness [10], while current state-of-the-art approaches are based on nonlinear model predictive control (NMPC), with hybrid, adaptive, and learning-based extensions [8], [9], [11], [12].

Despite these advances, existing cable-based systems only partially address autonomous pick-and-place. Magnetic grippers have been used to pick up static payloads [13], but they require an external mechanism for drop-off. Autonomous placement onto moving platforms has been considered in [14], but there the payload is pre-attached to the cable, so the controller does not need to solve the grasping problem. Hence, fully autonomous pick-and-place between moving platforms with a passive gripper has not yet been solved.

To combine the autonomous grasping ability of UAMs with the agility and energy efficiency of cable suspension, a hook-equipped aerial manipulator was introduced in [15] for static pick-and-place, and extended to dynamic grasping from a moving platform in [16]. In this design, a lightweight, passive hook is mounted on a rigid pole, which is connected to the quadrotor through an unactuated 2 DoF joint. While previous work on this platform demonstrated autonomous grasping and transportation, dynamic pick-and-place between two moving platforms has still remained a challenge.

In this work, we address these limitations by solving the following complex pick-and-place scenario. Initially, the payload with a hook on the top is carried by a moving ground platform. The quadrotor, equipped with a hook that connects to the drone

through a rigid pole and 2 DoF joint, has to autonomously approach and grasp the payload with its hook, transport it to the target location, and place it onto another moving platform by detaching its hook, all within prescribed grasping and placement time windows, while satisfying state and input constraints under uncertain payload mass.

From a control perspective, the introduced scenario poses several challenges. First, the coupled dynamics of the manipulator and the payload are highly nonlinear and include contact interactions that are difficult to capture accurately. Second, the system operates across multiple configurations (with and without attached payload), leading to abrupt changes in dynamic behavior. Third, uncertainty in the payload mass further affects the dynamics and must be handled in control design. Overall, achieving high performance requires sufficiently accurate models of these complex dynamics, and a control approach that effectively exploits these models.

To provide an efficient solution for this problem, we propose a novel NMPC with the following contributions:

- 1) To capture the complex dynamics of the transportation system, we use a MuJoCo-based [17] digital twin directly as the predictive model of the NMPC, offering a simple, accurate, and easily reconfigurable approach.
- 2) To handle the uncertainties of the predictive model due to uncertain payloads and external disturbances, we propose a robust adaptive NMPC that integrates zero-order uncertainty propagation with an extended Kalman filter for online parameter estimation, ensuring robust constraint satisfaction and high performance.
- 3) We propose an optimization-based feasibility analysis to compute admissible pick-and-place time windows and verify constraint satisfaction over a set of scenarios, providing quantitative timing and safety certificates.
- 4) We demonstrate the effectiveness of the proposed approach through extensive simulation studies and real-world flight experiments for aerial pick-and-place between two moving ground platforms.

II. MUJOCo MODEL FOR HOOK-BASED PICK-AND-PLACE

When describing the dynamics of an aerial transportation system, most works derive analytical models that rely on certain assumptions and simplifications, such as modeling the payload as a point mass, or assuming that the mass of the cable is negligible [8], [9], [11]. In contrast, we use the physics simulator MuJoCo to generate a digital twin model of the hook-based system for three key reasons: i) it simplifies the modeling procedure, as instead of analytical derivations, the elements can be put together by joints of different characteristics; ii) the resulting model is highly accurate, as the geometry and inertia of each component, including the hooks and the payload are directly considered; and iii) it is easy to reconfigure, as each element can be conveniently modified, if the design is changed (e.g. new hook geometry), and the corresponding model is auto-generated.

In modeling, we make one key simplification: to avoid discontinuities in the predictive model of the NMPC, we smoothly approximate the contact dynamics between the hook of the drone and the payload by augmenting the manipulator model with a 2 DoF revolute joint. Then, to address all phases of pick-and-place (with and without attached payload), we consider the mass of the payload as a parameter. During transportation, it equals to the actual mass value, otherwise it is set to zero to retrieve the dynamics of the manipulator. Later, we revisit the model

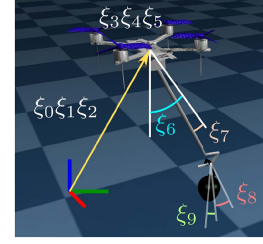


Fig. 2. MuJoCo model describing the hook-based aerial manipulator and its coupled dynamics with a payload.

mismatch corresponding to this approximation and integrate it into the robust MPC design, developed for flight control.

The resulting 10-DoF MuJoCo model with generalized coordinates ξ_0 – ξ_9 is illustrated in Fig. 2. The quadcopter has 3 translational and 3 rotational DoFs. Then, the pole is connected through two passive revolute joints. Finally, two rotational DoFs represent the attachment of the hooks.

Formally, the dynamic model is described as follows:

$$\xi_{k+1} = \phi(\xi_k, u_k, m_{L,k}), \quad (1)$$

where ξ denotes the 20-dimensional discrete-time state vector with the first ten variables corresponding to the generalized coordinates, while the rest are their time derivatives, and $u = [F \ \tau^\top]^\top$ is the control input vector containing the scalar collective thrust (F) and three-dimensional torque vector (τ) around the quadcopter body-fixed axes. The discrete-time state transition function is denoted by ϕ , the payload mass by m_L , and k is the discrete time index.

In this work, we assume that full state information is available for the controller. Measurements of the generalized coordinates can come from a motion capture system or encoders at the joints, while the velocity states can be either directly measured by inertial sensors or estimated by numerical differentiation of the position variables.

III. MPC DESIGN FOR DYNAMIC PICK-AND-PLACE

A. Overall Concept

Our control objective is to autonomously pick up a payload from a moving platform, transport it to another moving platform, and put it down. In addition, similar to [16], we incorporate scheduling constraints similar to the time management in industrial processes. Specifically, the quadcopter has to grasp and put down the payload within dedicated time windows. In terms of grasping, we impose the constraint $\underline{T}_g \leq T_g \leq \bar{T}_g$, and similarly for placement, $\underline{T}_p \leq T_p \leq \bar{T}_p$. The beginning and end of the time windows ($\underline{T}_g, \bar{T}_g, \underline{T}_p, \bar{T}_p$) are fixed in the problem description, and the exact grasping and placement times (T_g, T_p) are obtained from the actual solution of the NMPC algorithm. In addition, we assume that the future motion of the moving platforms are known, e.g., obtained by forward simulating their digital twins in terms of MuJoCo models (as done in [16]).

In most of the relevant literature, NMPC is used for tracking, and a separate trajectory planner is developed (see e.g. [8], [9], [11], [12]). In contrast, our proposed NMPC is used to design both the reference trajectory and the control inputs, providing a more holistic approach. Due to the fact that the goal is different in each phase of the task (grasping, transportation, and placement), we first define distinct motion phases with corresponding transition conditions. Then, we formulate a general optimal control

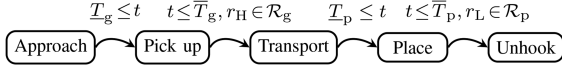


Fig. 3. Task phases and transition conditions.

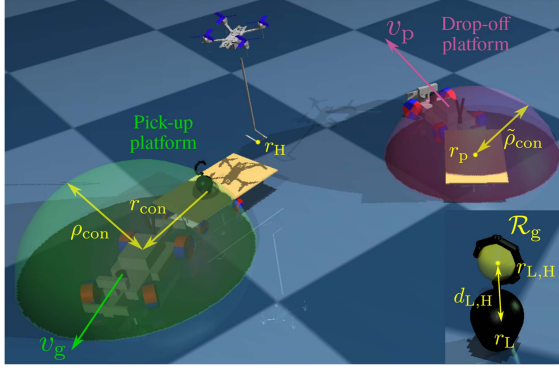


Fig. 4. Snapshot from MuJoCo simulation of the dynamic pick-and-place task, illustrating the moving platforms, the quadrotor with hook, and the position, velocity, and geometric quantities used in the MPC formulation.

problem (OCP), and describe the specific components of the OCP for each phase.

We divide the pick-and-place task to five motion phases:

- 1) *Approach*: Get close to the payload until $t \geq \underline{T}_g$
- 2) *Pick up*: Attach the hook to the payload before $\bar{T}_g$
- 3) *Transport*: Move the load towards the target until $t \geq \underline{T}_p$
- 4) *Place*: Put the payload on the target platform before $\bar{T}_p$
- 5) *Unhook*: Finish the task by detaching the hooks

The transitions between the phases are illustrated in Fig. 3. Phase 2 is activated when the grasping time window specified by external scheduling opens, i.e., $k \geq \underline{N}_g$, where $\underline{N}_g = \lfloor \underline{T}_g / \Delta t \rfloor$ with $\lfloor x \rfloor$ rounding to the nearest integer and Δt being the discretization time step.

To enter Phase 3, the hook needs to be attached. This condition is illustrated in Fig. 4 and formulated as follows:

$$r_{H,k} \in \mathcal{R}_g, \quad \mathcal{R}_g = \{r \in \mathbb{R}^3 \mid \|r - r_{L,H}\|_2 \leq \rho_H\}, \quad (2)$$

where r_H, r_L are the position of the hook and the center of mass of the payload, ρ_H is the radius of the hook on the payload, and $r_{L,H}$ is the position of the hook on the payload, computed as $r_{L,H} = r_L + R_L e_3 d_{L,H}$, where R_L is the rotation matrix of the payload and $e_3 = [0 \ 0 \ 1]^\top$.

Similar to Phase 2, Phase 4 starts when $k \geq \underline{N}_p$ with $\underline{N}_p = \lfloor \underline{T}_p / \Delta t \rfloor$. Finally, Phase 5 is activated when the distance of the payload and the drop-off platform gets below a threshold:

$$r_{L,k} \in \mathcal{R}_p, \quad \mathcal{R}_p = \{r \in \mathbb{R}^3 \mid \|r - r_p\|_2 \leq \varepsilon_p\}, \quad (3)$$

where r_p is the position of the drop-off platform and ε_p is a small constant, e.g., 1 mm.

B. Optimization Problem

We now introduce the generic form of the OCP, followed by the cost functions and constraints for each phase. Let $\xi = (\xi_0|k, \dots, \xi_N|k)$, $\mathbf{u} = (u_0|k, \dots, u_{N-1}|k)$ be the predicted states and chosen inputs at time k . The optimal trajectories are computed as the solution of the discrete-time OCP:

$$\min_{\xi, \mathbf{u}} \sum_{i=0}^{N-1} l(h_p(\xi_{i|k}), y_{p,k+i}^{\text{ref}}) + \|v_{i|k}\|_{W_v^p}^2 + \|u_{i|k}\|_{W_u}^2, \quad (4a)$$

$$\text{s.t. } \xi_{i+1|k} = \phi_p(\xi_{i|k}, u_{i|k}), \quad \forall i \in \mathbb{I}_0^{N-1}, \quad (4b)$$

$$\xi_0|k = \xi_k, \quad (4c)$$

$$\underline{\xi} \leq \xi_{i+1|k} \leq \bar{\xi}, \quad \underline{u} \leq u_{i|k} \leq \bar{u}, \quad \forall i \in \mathbb{I}_0^{N-1}, \quad (4d)$$

$$g_p(\xi_{i|k}) \leq 0, \quad \forall i \in \mathbb{I}_0^N, \quad (4e)$$

where $h_p(\xi_{i|k}), y_{p,k+i}^{\text{ref}}$ are the output and the output reference of the current phase, respectively, v denotes the velocity states of ξ , $\|\cdot\|_W$ stands for the weighted Euclidean norm with positive definite weight matrix W , and $p \in \mathbb{I}_1^5$ represents the current motion phase. At each time k , p is determined by the transition conditions of Fig. 3 and kept constant over the horizon. Although optimizing mid-horizon phase transitions may improve performance, we choose to keep it fixed for computational efficiency. This design choice is supported by the results of Section VI and VII, demonstrating high performance of the method. The dynamics, the initial conditions, and the state and input bounds are expressed by (4b)-(4d), while (4e) stands for nonlinear constraints on the state variables. The predictive model is set to $\phi_p(\cdot, \cdot) = \phi(\cdot, 0)$ for $p \in \{1, 2, 5\}$ and $\phi_p(\cdot, \cdot) = \phi(\cdot, \cdot, m_L)$ otherwise.

For stage cost, we use a smooth 1-norm approximation: $l(y_p, y_p^{\text{ref}}) = \sum_{j=0}^{n_y-1} w_{p,j} (\sqrt{(y_{p,j} - y_{p,j}^{\text{ref}})^2 + \gamma^2} - \gamma)$, where y_p, y_p^{ref} are the output and output reference of phase p , and $w_{p,j}, \gamma > 0$ are tuning parameters. Next, we introduce the specific cost terms and constraints for each phase.

Phase 1: To approach the payload, we penalize the difference of the position and orientation of the hook and the payload with the output and reference defined as follows:

$$h_1(\xi_{i|k}) = [r_{H,i|k}^\top \ \psi_{H,i|k}]^\top, \quad y_{1,k+i}^{\text{ref}} = [r_{L,H,k}^{\text{ref}\top} \ \psi_{L,k}^{\text{ref}}]^\top, \quad (5)$$

where ψ denotes the yaw angle of the hook and the payload. To make sure that the payload is not grasped for $k \leq \underline{N}_g$, we introduce the following constraint:

$$g_1(\xi_{i|k}) = \rho_{\text{con}} - \|r_{H,i|k} - (r_{L,H,k+i}^{\text{ref}} + R_{L,k+i}^{\text{ref}} r_{\text{con}})\|_2. \quad (6)$$

Constraint (6) ensures that the position of the hook is outside of a sphere with radius $\rho_{\text{con}} \in \mathbb{R}^+$ and center point $r_{L,H,k+i}^{\text{ref}} + R_{L,k+i}^{\text{ref}} r_{\text{con}}$, where $r_{\text{con}}, \rho_{\text{con}}$ are tunable parameters. This is illustrated in Fig. 4, where the green shaded area represents the set that has to be avoided by the hook of the quadcopter.

Phase 2: Next, the objective is to grasp the payload, which is achieved by using the following output and reference:

$$h_2(\xi_{i|k}) = h_1(\xi_{i|k}), \quad y_{2,k+i}^{\text{ref}} = [r_{L,H,k+i}^{\text{ref}\top} \ \psi_{L,k+i}^{\text{ref}}]^\top. \quad (7)$$

In this phase, nonlinear constraints are not imposed.

Phase 3: Once the hook is attached to the payload, the goal is to move the load to another moving vehicle (drop-off platform) with predicted position r_p^{ref} and yaw angle ψ_p^{ref} . For this, the output and reference are defined as follows:

$$h_3(\xi_{i|k}) = [r_{L,i|k}^\top \ \psi_{L,i|k}]^\top, \quad (8a)$$

$$y_{3,k+i}^{\text{ref}} = [(r_{p,k+i}^{\text{ref}} + z_{\text{safe},k} e_3)^\top \ \psi_{p,k+i}^{\text{ref}}]^\top. \quad (8b)$$

Notice that the payload position is now a decision variable, not an external signal, therefore the ref superscript is omitted. In (8b), $e_3 = [0 \ 0 \ 1]^\top$, and $z_{\text{safe},k} > 0$ is used to ensure that the payload is transported above the moving platform before putting it down.

In this phase, similar to Phase 1, we introduce nonlinear constraints to prevent putting down the payload for $k \leq \underline{N}_p$:

$$g_{3,m+1}(\xi_{i|k}) = \tilde{\rho}_{\text{con}} - \|r_{L,i|k} - r_{p,k+i}^{\text{ref}}\|_2. \quad (9)$$

Constraint (9) ensures that the position of the transported payload is outside of a sphere with radius $\tilde{\rho}_{\text{con}} \in \mathbb{R}^+$ centered around the target moving platform.

Phase 4: In this phase, the objective is similar to the previous one, however, instead of only moving towards the target, the aim is to put down the payload. Hence, the controlled output and the reference are the same as in Phase 3, $h_4(\xi_{i|k}) = h_3(\xi_{i|k})$, $y_{4,k+i}^{\text{ref}} = y_{3,k+i}^{\text{ref}}$, but (9) is no longer active, making the placement possible.

Phase 5: The objective of the last phase is to detach the hook from the payload:

$$h_5(\xi_{i|k}) = \begin{bmatrix} r_{H,i|k}^\top & \psi_{H,i|k} \end{bmatrix}^\top, \quad (10a)$$

$$y_{5,k+i}^{\text{ref}} = \begin{bmatrix} (r_{p,k+i}^{\text{ref}} + \bar{x}R_{p,k+i}^{\text{ref}}e_1)^\top & \psi_{p,k+i}^{\text{ref}} \end{bmatrix}^\top, \quad (10b)$$

where $e_1 = [1 \ 0 \ 0]^\top$ and $\bar{x} \in \mathbb{R}^+$ is a parameter that determines the setpoint of the hook. Its numerical value is chosen to be greater than the radius of the hook, e.g., 0.3 m.

IV. ROBUST ADAPTIVE MPC (RAMPC)

A. Problem Formulation and Uncertain Dynamics

In real-world aerial transportation, it is realistic to assume that the parameters of the manipulator are reasonably well known. However, there still remain significant model uncertainties, e.g. due to the contact dynamics between the hooks that we neglected in the predictive model (see Section II), complex aerodynamics, and payloads with uncertain mass.

In this section, we introduce a robust adaptive MPC approach that is able to handle both parametric uncertainties and external disturbances. For this, first, we introduce the considered uncertain dynamic model. Then, we show how zero-order robust optimization (zoRO) [18] can be applied to solve the resulting OCP with minimal additional computational effort compared to the solution of (4). Finally, we describe how an extended Kalman filter (EKF) can be used to estimate uncertain parameters online and systematically improve the performance of the robust approach.

Let $\mathbb{S}_+^n$ denote the set of symmetric, positive definite matrices of dimension n . Then, for any $q \in \mathbb{R}^n$, $Q \in \mathbb{S}_+^n$, an ellipsoid is defined as $\mathcal{E}(q, Q) = \{x \in \mathbb{R}^n \mid \|x - q\|_{Q^{-1}} \leq 1\}$. We extend the predictive model (4b) with parametric uncertainty and external disturbance as follows:

$$\xi_{k+1} = \phi(\xi_k, u_k, m_{L,k}, \theta) + w_k = \phi_p(\xi_k, u_k, \theta) + w_k, \quad (11)$$

where $\phi_p(\cdot, \cdot, \cdot) = \phi(\cdot, \cdot, 0, \cdot)$ for $p \in \{1, 2, 5\}$ and $\phi_p(\cdot, \cdot, \cdot) = \phi(\cdot, \cdot, m_L, \cdot)$ otherwise. Additionally, $\theta \in \mathcal{E}(\bar{\theta}, W_\theta)$, $\bar{\theta} \in \mathbb{R}^{n_\theta}$, $W_\theta \in \mathbb{S}_+^{n_\theta}$ is the vector of uncertain parameters, and $w_k \in \mathcal{E}(0, W_w)$, $W_w \in \mathbb{S}_+^{n_w}$ comprises unmodeled dynamics.

B. zoRO With Ellipsoidal Uncertainty Sets

Zero-order robust optimization (zoRO) [18], [19] achieves computational efficiency by (i) approximating the true invariant set of the nonlinear system with ellipsoids and (ii) decoupling the computation of constraint tightening from the optimization, resulting in substantial dimensionality reduction. Next, based

on [18], [19], we describe a robustified version of the OCP in (4), followed by the zoRO construction.

To extend the OCP formulation in (4), let us describe the predicted ellipsoidal uncertainty sets by $\Sigma = (\Sigma_{0|k}, \dots, \Sigma_{N|k})$, centered at $\bar{\xi} = (\bar{\xi}_{0|k}, \dots, \bar{\xi}_{N|k})$. Then, we formulate the following robustified OCP:

$$\min_{\bar{\xi}, u, \Sigma} \sum_{i=0}^{N-1} l(\bar{\xi}_{i|k}, u_{i|k}), \quad (12a)$$

$$\text{s.t. } \bar{\xi}_{i+1|k} = \phi_p(\bar{\xi}_{i|k}, u_{i|k}, \bar{\theta}), \quad \forall i \in \mathbb{I}_0^{N-1} \quad (12b)$$

$$\Sigma_{i+1|k} = \Phi(\Sigma_{i|k}, W, \bar{\xi}_{i|k}, u_{i|k}), \quad \forall i \in \mathbb{I}_0^{N-1} \quad (12c)$$

$$\bar{\xi}_{0|k} = \xi_k, \quad \Sigma_{0|k} = \bar{\Sigma}, \quad (12d)$$

$$g(\bar{\xi}_{i|k}, u_{i|k}) + \beta(\bar{\xi}_{i|k}, u_{i|k}, \Sigma_{i|k}) \leq 0, \quad \forall i \in \mathbb{I}_0^{N-1} \quad (12e)$$

where $W = \text{diag}(W_\theta, W_w)$, g is the stacking of (4d)-(4e), and $\bar{\Sigma}$ represents uncertainty in the measured state such that $\xi_k \in \mathcal{E}(\bar{\xi}_k^{\text{true}}, \bar{\Sigma})$. The backoff terms of the constraint tightening, denoted by β , are computed as follows:

$$\begin{aligned} \beta_j(\bar{\xi}_{i|k}, u_{i|k}, \Sigma_{i|k}) \\ = \sqrt{\nabla_\xi g_j(\bar{\xi}_{i|k}, u_{i|k})^\top \Sigma_{i|k} \nabla_\xi g_j(\bar{\xi}_{i|k}, u_{i|k})} \quad \forall j \in \mathbb{I}_1^{n_g}. \end{aligned} \quad (13)$$

The ellipsoidal uncertainty sets are propagated through the dynamics with the transition function Φ defined as follows:

$$\Phi(\Sigma, W, \xi, u) = A(\xi, u)\Sigma A(\xi, u)^\top + G(\xi, u)WG(\xi, u)^\top, \quad (14)$$

where A, G denote the Jacobians of the dynamics w.r.t the states and uncertainties:

$$A(\xi, u) = \frac{\partial \phi_p}{\partial \xi} \Big|_{\xi, u, \bar{\theta}}, \quad G(\xi, u) = \begin{bmatrix} \frac{\partial \phi_p}{\partial \theta} \Big|_{\xi, u, \bar{\theta}} & I_{n_w} \end{bmatrix}. \quad (15)$$

The uncertainty propagation (14) is a standard approximation that has already proven to be a good tradeoff between accuracy and computational load [18].

As it is explained in [18], although (12) can be solved with standard nonlinear programming, the number of optimization variables is significantly increased compared to the nominal OCP given by (4). Hence, another approximation is introduced by computing the backoff terms using the previous solution and fixing them during the optimization. This way, the following zoRO problem is introduced:

$$\min_{\bar{\xi}, u, \Sigma} \sum_{i=0}^{N-1} l(\bar{\xi}_{i|k}, u_{i|k}), \quad (16a)$$

$$\text{s.t. } \bar{\xi}_{i+1|k} = \phi_p(\bar{\xi}_{i|k}, u_{i|k}, \bar{\theta}), \quad \forall i \in \mathbb{I}_0^{N-1}, \quad (16b)$$

$$\bar{\xi}_{0|k} = \xi_k, \quad (16c)$$

$$g(\bar{\xi}_{i|k}, u_{i|k}) + \hat{\beta}_i \leq 0, \quad \forall i \in \mathbb{I}_0^{N-1}, \quad (16d)$$

where $\hat{\beta}_i$ are approximate backoff terms, recomputed at each iteration of the optimizer by (i) linearizing around the current nominal trajectory, (ii) propagating the uncertainties from initial condition $\bar{\Sigma}$ with (14), and (iii) applying (13). The first input $u_{0|k}$ is applied in a receding horizon fashion.

C. Extended Kalman Filter for Parameter Estimation

Robust MPC ensures constraint satisfaction under uncertain parameters and bounded model mismatch. Next, we introduce

an extended Kalman filter (EKF) for online parameter estimation to improve tracking, estimate the uncertainty of the parameters, and reduce conservativeness. Later in experiments, EKF is used to estimate the uncertain payload mass.

To adapt Kalman filtering for the task at hand, the uncertain dynamics (11) have to be revised. Specifically, we assume that the uncertainty $w_k \in \mathcal{E}(0, W_w)$ can be described by a normal distribution $\tilde{w}_k \sim \mathcal{N}(0, \Sigma_w)$. Then, we compute the covariance matrix based on the assumption that $\mathcal{E}(0, W_w)$ corresponds to the confidence region, as follows:

$$w_k^\top W_w^{-1} w_k \leq 1 \Leftrightarrow \tilde{w}_k^\top \Sigma_w^{-1} \tilde{w}_k \leq \chi_{n_w, \alpha}^2 \quad (17)$$

where $\chi_{n_w, \alpha}^2$ is the value of the inverse CDF of the chi-squared distribution with n_w degrees of freedom for probability α . From (17), Σ_w is expressed as

$$\Sigma_w = W_w / \chi_{n_w, \alpha}^2. \quad (18)$$

This way, the underlying model of the parameter estimator is

$$\bar{\theta}_{k+1} = \bar{\theta}_k + \eta_k, \quad y_k = d(\xi_k, u_k, \bar{\theta}, e_k) \quad (19)$$

where $\eta_k \sim \mathcal{N}(0, Q)$, $Q \in \mathbb{S}_+^{n_\theta}$ is an artificial, small white noise input that controls the rate of the parameter change, while $e_k \sim \mathcal{N}(0, R)$, $R \in \mathbb{S}_+^{n_\epsilon}$ models the measurement noise and the model uncertainty. The output is defined as

$$\begin{aligned} y_k &= d(\xi_k, u_k, \bar{\theta}, e_k) = x_k + e_k \\ &= \phi_p(\xi_{k-1}, u_{k-1}, \bar{\theta}) + \tilde{w}_{k-1} + e_k. \end{aligned} \quad (20)$$

With the model and output equations in place, the parameter estimate $\hat{\theta}_k$ and the covariance P_k can be computed using the standard EKF update equations, described e.g. in [20]. Note that EKF can only estimate constant or slowly varying parameters. If significant unmodeled dynamics are present, reliable adaptation requires recursive identification or learning methods that are computationally more expensive.

D. Integration of EKF Covariance and Uncertainty Bounds

So far, we introduced two extensions to the nominal MPC: uncertainty propagation by zoRO and parameter estimation by EKF. Next, we derive a systematic integration of these two components by matching the covariance of the EKF estimate with the uncertainty bounds of the robust optimization.

In the robust OCP (12), uncertain parameters are described by the ellipsoid $\theta \in \mathcal{E}(\bar{\theta}, W_\theta)$. The estimated parameters by the EKF are tracked with a normal distribution $\theta_k \sim \mathcal{N}(\hat{\theta}_k, P_k)$. Similar to (17)–(18), we match the confidence region of the normal distribution with the uncertainty bounds to arrive at the following update law:

$$W_{\theta, k} = \chi_{n_\theta, \alpha}^2 P_k. \quad (21)$$

By selecting the confidence level α , the volume and reliability trade-off of the safety margin can be controlled by the user (e.g., 95% is a standard choice).

Each time an EKF update is computed, the current estimate $\hat{\theta}_k$ is used to evaluate the dynamics and the Jacobians in Optimization (16) and the covariance P_k is used to update the uncertainty bound by (21), such that $W = \text{blkdiag}(W_{\theta, k}, W_w)$. With this, we have a systematic integration of uncertainty propagation and parameter estimation.

V. ADMISSIBLE TIME WINDOWS AND FEASIBILITY ANALYSIS

In this section, we propose an optimization-based approach to analyze the feasibility of the proposed method over a set of admissible scenarios. For this, we aim to determine *admissible time*

windows, parametrized by $\underline{T}_g, \bar{T}_g, \underline{T}_p, \bar{T}_p$, such that RAMPC is feasible, all constraints are satisfied, and the pick-and-place is successful for every scenario in the considered sets. To address this in a computationally tractable way, we make two assumptions: i) the task is executed in a fixed flight space with known bounds, and ii) a parametrization is available that describes the trajectories of the ground platforms. In automated logistics or manufacturing systems, these assumptions are realistic, since indoor spaces are bounded and conveyor belts or mobile robots move along well-defined paths.

A. Admissible Grasping Time Windows

We first provide an algorithm to compute admissible pick-up time windows $[\underline{T}_g, \bar{T}_g]$. The main challenges are that i) if $\underline{T}_g$ is too small, the manipulator does not have enough time to start tracking the moving platform, and ii) if $\bar{T}_g - \underline{T}_g$ is too small, there is not enough time for grasping.

Scenarios are characterized by a parameter vector η that describes the initial position of the quadcopter and the trajectory of the moving platforms, and comes from a bounded set of admissible parameters $\mathcal{H}$. For a fixed time window $[\underline{T}_g, \bar{T}_g]$ and a given scenario η , we run closed-loop simulations with RAMPC and define the grasping time $T_g(\eta, \underline{T}_g, \bar{T}_g)$ as the first point in time for which the grasping condition (2) is satisfied. We say that RAMPC is *feasible* for scenario η and time window $[\underline{T}_g, \bar{T}_g]$ if it remains feasible at all time steps, all constraints are satisfied, and $\underline{T}_g \leq T_g(\eta, \underline{T}_g, \bar{T}_g) \leq \bar{T}_g$. We call a window $[\underline{T}_g, \bar{T}_g]$ *admissible* over $\mathcal{H}$ if RAMPC is feasible for all $\eta \in \mathcal{H}$.

To compute admissible grasping windows, we assume that the scenario set $\mathcal{H}$ and a user-defined upper bound $\bar{T}_g^{\max}$ with $\bar{T}_g \leq \bar{T}_g^{\max}$ are available. As a first step, we set $\bar{T}_g = \bar{T}_g^{\max}$ and choose a $\underline{T}_g = \bar{T}_g - \varepsilon_T$, where ε_T is a small positive number (e.g., 0.1s). For this candidate window, we run simulations for all $\eta \in \mathcal{H}$ and compute the *worst-case grasping time*

$$\bar{T}_g^* = \max_{\eta \in \mathcal{H}} T_g(\eta, \underline{T}_g, \bar{T}_g), \quad (22)$$

provided all scenarios remain feasible.

If there are infeasible scenarios (e.g., because $\bar{T}_g - \underline{T}_g$ is too small), we decrease $\underline{T}_g$ to enlarge the window until RAMPC is feasible for all $\eta \in \mathcal{H}$. If feasibility cannot be restored for any $\underline{T}_g \leq \bar{T}_g^{\max}$, then no admissible window exists for the given $(\bar{T}_g^{\max}, \mathcal{H})$. In this case, the user must either increase $\bar{T}_g^{\max}$ or restrict $\mathcal{H}$. Once feasibility is ensured, we use bisection to find the smallest feasible $\underline{T}_g$. This smallest feasible lower bound is denoted by $\underline{T}_g^*$, and the corresponding worst-case grasping time by $\bar{T}_g^* = \max_{\eta \in \mathcal{H}} T_g(\eta, \underline{T}_g^*, \bar{T}_g^{\max})$. As a result of the algorithm, we obtain that any admissible grasping window must satisfy $\underline{T}_g^* \leq \underline{T}_g$ and $\bar{T}_g^* - \underline{T}_g^* \leq \bar{T}_g - \underline{T}_g$, i.e., $\underline{T}_g^*$ and $\bar{T}_g^* - \underline{T}_g^*$ are the lower bounds on the grasping start time and on the window width, respectively.

Solving (22) analytically would be extremely challenging, as each evaluation requires simulating a full scenario with RAMPC. Instead, we compute an approximation of the worst-case grasping time with Bayesian Optimization (BO) that systematically selects and evaluates scenarios in $\mathcal{H}$ to optimize (22) for a given candidate window. To additionally verify constraint satisfaction for a given window, we compute the *worst-case*

constraint violation

$$\nu_g^* = \max_{\eta \in \mathcal{H}} \max_{k \in \mathbb{I}_0^{N_g}} g(\xi_k^\eta, u_k^\eta), \quad (23)$$

where g is the constraint vector from (16d) and (ξ_k^η, u_k^η) denotes the closed-loop trajectory for scenario η . If $\nu_g^* \leq 0$, the window $[\underline{T}_g, \bar{T}_g]$ is admissible over $\mathcal{H}$. Otherwise, the scenario set $\mathcal{H}$ or the parameters of the RAMPC must be adjusted. The worst-case value of ν_g^* is approximately computed by BO, analogous to the solution of (22).

B. Admissible Placement Time Windows

An analogous procedure is used to determine admissible placement windows $[\underline{T}_p, \bar{T}_p]$. Here, the goal is to ensure that the payload can be transported from the pick-up configuration to the drop-off platform and placed within the time interval, for a set of scenarios $\mathcal{M}$. Scenarios are characterized by a parameter vector μ that includes the initial configuration, the trajectories of the moving platforms, and the payload mass. For fixed $[\underline{T}_p, \bar{T}_p]$ and μ , we run closed-loop simulations with RAMPC (assuming the payload is already attached) and define the drop-off time $T_p(\mu, \underline{T}_p, \bar{T}_p)$ as the first point in time for which the drop-off condition (3) is satisfied. We use the same terminology as before: RAMPC is feasible for μ and $[\underline{T}_p, \bar{T}_p]$ if it remains feasible, satisfies all constraints, and achieves $\underline{T}_p \leq T_p(\mu, \underline{T}_p, \bar{T}_p) \leq \bar{T}_p$; the window is admissible over $\mathcal{M}$ if this holds for all $\mu \in \mathcal{M}$.

From preliminary simulations, we observe that successful placement does not require tracking of the target platform, hence, we fix $\underline{T}_p = 0$ and search for the smallest admissible $\bar{T}_p$. For this, we specify a user-defined upper bound $\bar{T}_p^{\max}$ with $\bar{T}_p \leq \bar{T}_p^{\max}$ and compute the *worst-case drop-off time*

$$\bar{T}_p^* = \max_{\mu \in \mathcal{M}} T_p(\mu, 0, \bar{T}_p^{\max}), \quad (24)$$

using BO. Finally, to certify constraint satisfaction and feasibility for all $\mu \in \mathcal{M}$ with a candidate $\bar{T}_p^*$, we compute the worst-case constraint violation in the transportation phase

$$\nu_p^* = \max_{\mu \in \mathcal{M}} \max_{k \in \mathbb{I}_0^{N_p}} g(\xi_k^\mu, u_k^\mu), \quad (25)$$

where the notations are analogous to (23). If $\nu_p^* \leq 0$, the placement window $[0, \bar{T}_p^*]$ is admissible over $\mathcal{M}$, otherwise either $\mathcal{M}$ or the RAMPC parameters must be adjusted. Finally, as a result of the algorithm, we obtain a lower bound $\bar{T}_p^*$ on the width of the placement time window.

VI. SIMULATION STUDY

A. Test Environment

In our test arena, the flight space is a $7 \times 5 \times 3$ m rectangle. To carry the payload, we use F1Tenth ground vehicles [21] with custom designed trailers. For the simulation study, the digital twin of each component has been implemented in MuJoCo as illustrated in Fig. 4. Throughout the analyzed scenarios, the ground platforms are moving along paperclip-shaped trajectories with constant speed. These trajectories are characterized by two parameters: the starting point along this periodic path ($s_g \in [0, 1]$) and the speed of the towing vehicle (v_g for the pick-up platform, v_p for the drop-off platform).

We run the MPC on a desktop PC with Intel i9 CPU. For the numerical solution, the SQP-RTI solver of *acados* is used [22].

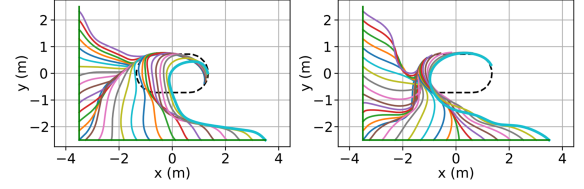


Fig. 5. (x, y) position trajectories of the quadcopter before grasping the payload (left) and during transportation (right). The black dashed line shows the path of the moving platform, while the green line shows the set of initial positions when solving (22) and (24). The bold cyan curve highlights the worst-case trajectory in terms of pick-up and drop-off time, respectively.

The Jacobians of the MuJoCo dynamics are computed with finite differences (FD), and the predictive model is discretized with the *implicit-in-velocity Euler* integrator of MuJoCo and with sampling time $\Delta t = 0.05$ s. The FD Jacobian computation is chosen instead of the automatic differentiation of MuJoCo-MJX¹ due to the fact that for the model described in Section II, the difference between the output of the two algorithms is near floating-point precision, and the FD approach is computationally more efficient for this model. The state measurement uncertainty bound is set to $\bar{\Sigma} = 10^{-4} I_{n_\xi}$ and the EKF covariance is initialized by $P_0 = 10^{-2} I_{n_\theta}$. The horizon length is $N = 25$, and the solution is recomputed with a control frequency of 50 Hz. In simulation, this proven to be sufficient, and, in real experiments, the computed body rates are sent to the quadcopter and tracked by a low-level PD controller that runs onboard at 500 Hz. To solve (22)–(25) numerically, we use BoTorch [23].

As explained in Section II, the predictive model of NMPC relies on a smoothened hook–payload interaction. In contrast, the main simulation loop uses the internal contact model of MuJoCo. This induces a deliberate mismatch between these models, so that the robustness of the proposed controller can be evaluated against more realistic contact dynamics.

B. Admissible Time Windows

Next, the time window parameters are optimized to make sure the pick-and-place will be successful under the available flight space and moving platform trajectory. The scenario parameter vector is defined as $\eta = [p_{xy} \ s_g \ v_g] \in \mathbb{R}^3$. Here, $p_{xy} \in [0, 1]$ is introduced to describe the set of initial positions for the quadcopter with a single scalar variable. The (x, y) initial positions are fixed to the boundary of the flight space, shown by the green lines in Fig. 5, to reduce the dimensionality of η , and the initial z coordinate is fixed to 1m. The mapping $p_{xy} \rightarrow (x, y)$ is defined as follows:

$$(x, y) = \begin{cases} (-3.5, 2.5 - 10p_{xy}), & \text{if } p_{xy} \in [0, 0.5] \\ (-3.5 + 14(p_{xy} - 0.5), -2.5), & \text{if } p_{xy} \in (0.5, 1]. \end{cases}$$

Further, $s_g \in [0, 1]$ admits the moving platform to start from any point of the periodic path, and $v_g \in [0.4, 0.6]$ m/s is the permitted speed of the moving platform.

The computed worst-case admissible grasping time windows, see Section V-A, are $\underline{T}_g^* = 4.3$ s and $\bar{T}_g^* = 7.7$ s, with $\eta^* = [1.0 \ 0.0 \ 0.6]$. This means that if the start of the grasping time window is $\underline{T}_g \geq 4.3$ s and its duration is $\bar{T}_g - \underline{T}_g \geq 3.4$ s, then the manipulator has enough time to start tracking the moving platform in Phase 1 and successfully pick up the payload in

¹<https://mujoco.readthedocs.io/en/stable/mjx.html>

TABLE I
PERFORMANCE OF NOMINAL AND RAMPC IN SIMULATION.

Mass deviation	0%		10%		20%		50%	
Controller	Nom	RA	Nom	RA	Nom	RA	Nom	RA
Success rate (%)	92	100	91	100	86	100	39	100
Average cost	4.98	6.47	5.17	6.53	5.19	6.54	5.16	6.63

Phase 2. Simulated trajectories of the quadcopter are displayed in Fig. 5, showing that $T_g = 4.3$ s is indeed enough to start tracking the moving payload from any initial position along the boundaries of the flight space. The worst-case trajectory is indicated by the bold cyan curve.

Next, we look for the feasible set of time windows for the transportation of a payload with uncertain mass. In this case, the scenario parameter vector is $\mu = [p_{xy} \ s_p \ v_p \ m_L] \in \mathbb{R}^4$. The admissible sets of the first three parameters are the same as previously, and the payload mass is restricted to $m_L \in [0.05, 0.2]$ kg. Its nominal value (used to initialize the EKF) is fixed to $m_L^0 = 0.07$ kg. The numerical result of solving (24) with BO is $T_p^* = 10.28$ s with $\mu^* = [1.0 \ 0.29 \ 0.6 \ 0.05]$. This means that the worst-case time needed to transport the payload to the target platform under the considered assumptions is 10.28 s. The right panel of Fig. 5 shows simulated trajectories of the quadcopter during transportation, with the worst-case scenario μ^*, T_p^* highlighted by the bold cyan curve.

With this, we have successfully computed the worst-case time window parameters under the considered assumptions, and have shown that even for these scenarios, the dynamic pick-and-place is always accomplished.

C. Comparison of Nominal and RAMPC in Simulations

To show the benefits of the robust adaptive extension, we now provide a comparison with the nominal MPC that is described in Section III. For this, we run simulations by randomizing the scenario parameters $\mu = [p_{xy} \ s_g \ v_g \ m_L]$. Moreover, to assess the performance of the controllers under a mismatch between the nominal and actual payload mass values, we simulate all scenarios with 0%, $\pm 10\%$, $\pm 20\%$, $\pm 50\%$ difference between these values. The results of simulating 800 scenarios overall are shown in Table I. The proposed RAMPC achieves a 100% success rate in all cases. In contrast, nominal MPC loses feasibility in some scenarios even when the nominal and actual payload mass coincide, resulting in 92% success. With increasing mass deviation, the success rate slightly decreases, until it significantly drops at 50% deviation.

Looking at the average costs, we see that the cost achieved by RAMPC is around 25% higher than the baseline. This is due to the conservativeness of the uncertainty propagation of zoRO. However, this performance tradeoff is justified by the 100% success rate over all examined scenarios.

The computation times of the nominal and RAMPC are displayed in the right panel of Fig. 7, showing that the proposed approach requires only an additional 1.5 ms (both in average and worst-case) compared to the baseline, making it real-time feasible at 50 Hz. The motion trajectory of a simulated pick-and-place scenario is shown in Fig. 6.

VII. FLIGHT EXPERIMENTS

We evaluate the real-world performance of the RAMPC in flight experiments. Our quadrotor is equipped with a Crazyflie

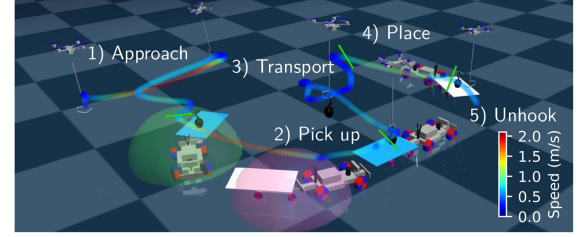


Fig. 6. Dynamic pick-and-place in simulation by RAMPC. The trajectory of the hook of the aerial manipulator is displayed in colors, while the boundaries of the motion phases are indicated with green dashes.

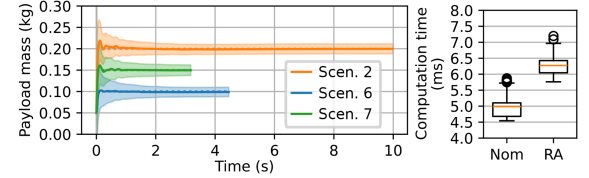


Fig. 7. Left: payload mass estimation by RAMPC in simulations, with $\pm 3\sigma$ interval corresponding to the EKF covariance. The horizontal axis shows $t \in [0, T_p - T_g]$. Right: computation time of nominal and RAMPC.

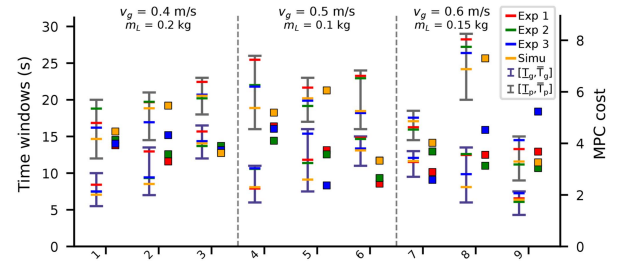


Fig. 8. RAMPC performance across 9 scenarios (x -axis) in real flights and simulation. The bars show the chosen pick-up and drop-off time windows, colored dashes indicate the actual pick-up and drop-off times, while squares display the achieved cost by the MPC for each trial.

Bolt flight controller² that receives body rate commands from the PC and runs a low-level PD controller at 500 Hz, using the on-board IMU measurements. Optitrack is used to provide high-precision pose information.

For performance evaluation, we consider 9 representative scenarios, grouped by the speed of the moving platforms $v_{g,p} \in \{0.4, 0.5, 0.6\}$ m/s and the payload mass $m_L \in \{0.2, 0.1, 0.15\}$ kg, respectively. Within each group, we vary the time windows, and the initial position of the quadrotor along the line p_{xy} in Section VI-B. For each scenario, we perform 3 real-world trials, resulting in 27 flight experiments overall. Fig. 8 shows the time windows, the achieved pick-up and drop-off times, and the MPC cost for each scenario, compared against the corresponding simulated baseline. RAMPC achieves a 100% success rate across all trials with consistent performance both across repeated trials and between simulation and real experiments. In particular, both the achieved pick-up and drop-off times and the resulting MPC cost are comparable between simulation and real experiments, confirming the sim-to-real applicability of the approach.

²<https://www.bitcraze.io/products/crazyflie-bolt-1-1/>

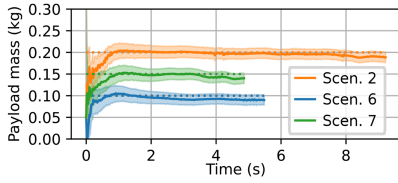


Fig. 9. Payload mass estimation by RAMPC in flight experiments, with $\pm 3\sigma$ interval corresponding to the EKF covariance. The horizontal axis shows $t \in [0, T_p - T_g]$, where the EKF is active.

To further evaluate the online parameter estimation, we highlight one representative scenario from each of the three parameter groups described above. These scenarios are labeled 2, 6, and 7 in Fig. 8. The left panel of Fig. 7 shows the convergence of the EKF estimate to the actual payload mass in simulation, while Fig. 9 shows the corresponding result in the real flight experiments. In both cases, the estimate remains within the $\pm 3\sigma$ confidence interval, and converges close to the actual mass value, although convergence in the real experiments is slightly slower (~ 1 s) than in simulation, due to unmodeled sensor noise and disturbances.

Fig. 1 shows the measured trajectory of the hook of the quadrotor for a representative flight experiment. The motion phases can be clearly identified: 1) tracking the pick-up platform and maintaining the speed $v_g = 0.5$ m/s, 2) accelerating to attach the hooks, 3) tracking the drop-off platform, 4) putting down the payload and 5) detaching the hooks. Video recordings of various flight experiments are included in the supplementary material.

VIII. CONCLUSION

This letter presented a robust adaptive nonlinear MPC approach for autonomous pick-and-place between moving platforms using a hook-equipped aerial manipulator. By integrating zero-order robust optimization and EKF-based online parameter estimation into a digital twin-based MPC, robust constraint satisfaction and high performance under uncertain payload mass is achieved. An optimization-based feasibility analysis further computes admissible grasping and placement time windows and provides safety certificates over a scenario set. Extensive simulations and real-world flight experiments demonstrate safe and precise payload transportation with consistent performance across significant payload mass variations.

Future work aims to solve fully onboard deployment, e.g. using vision-based sensing to eliminate reliance on external sensors. Moreover, providing formal recursive feasibility and stability guarantees of the closed-loop system would be a highly valuable addition to the proposed method.

ACKNOWLEDGMENT

The authors would like to thank Johannes Köhler and Amon Lahr for their valuable discussions and feedback.

REFERENCES

- [1] F. Ruggiero, V. Lippiello, and A. Ollero, "Aerial manipulation: A literature review," *IEEE Robot. Automat. Lett.*, vol. 3, pp. 1957–1964, Jul. 2018.
- [2] A. Ollero, M. Tognon, A. Suarez, D. Lee, and A. Franchi, "Past, present, and future of aerial robotic manipulators," *IEEE Trans. Robot.*, vol. 38, no. 1, pp. 626–645, Jan. 2022.

- [3] W. Luo, J. Chen, H. Ebel, and P. Eberhard, "Time-optimal handover trajectory planning for aerial manipulators based on discrete mechanics and complementarity constraints," *IEEE Trans. Robot.*, vol. 39, no. 6, pp. 4332–4349, Dec. 2023.
- [4] S. Wang, Z. Ma, F. Quan, and H. Chen, "Impact absorbing and compensation for heavy object catching with an unmanned aerial manipulator," *IEEE Robot. Automat. Lett.*, vol. 9, no. 4, pp. 3656–3663, Apr. 2024.
- [5] A. Kumar and L. Behera, "Thrust microstepping via acceleration feedback in quadrotor control for aerial grasping of dynamic payload," *IEEE Robot. Automat. Lett.*, vol. 9, no. 2, pp. 1246–1253, Feb. 2024.
- [6] G. Li, A. Tuncel, and G. Loianno, "PCMP: Perception-constrained model predictive control for quadrotors with suspended loads using a single camera and IMU," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2021, pp. 2012–2018.
- [7] S. Sun, X. Wang, D. Sanalitra, A. Franchi, M. Tognon, and J. Alonso-Mora, "Agile and cooperative aerial manipulation of a cable-suspended load," *Sci. Robot.*, vol. 10, no. 107, 2025, Art. no. eadu8015.
- [8] H. Li, H. Wang, C. Feng, F. Gao, B. Zhou, and S. Shen, "AutoTrans: A complete planning and control framework for autonomous UAV payload transportation," *IEEE Robot. Automat. Lett.*, vol. 8, no. 10, pp. 6859–6866, Oct. 2023.
- [9] H. Wang, H. Li, B. Zhou, F. Gao, and S. Shen, "Impact-aware planning and control for aerial robots with suspended payloads," *IEEE Trans. Robot.*, vol. 40, pp. 2478–2497, 2024.
- [10] K. Sreenath, T. Lee, and V. Kumar, "Geometric control and differential flatness of a quadrotor UAV with a cable-suspended load," in *Proc. IEEE Conf. Decis. Control*, 2013, pp. 2269–2274.
- [11] M. Sarvaiya, G. Li, and G. Loianno, "HPA-MPC: Hybrid perception-aware nonlinear model predictive control for quadrotors with suspended loads," *IEEE Robot. Automat. Lett.*, vol. 10, no. 1, pp. 358–365, Jan. 2025.
- [12] F. Panetsos, G. C. Karras, and K. J. Kyriakopoulos, "GP-based NMPC for aerial transportation of suspended loads," *IEEE Robot. Automat. Lett.*, vol. 10, no. 1, pp. 524–531, Jan. 2025.
- [13] S. Belkhal, R. Li, G. Kahn, R. McAllister, R. Calandra, and S. Levine, "Model-based meta-reinforcement learning for flight with suspended payloads," *IEEE Robot. Automat. Lett.*, vol. 6, no. 2, pp. 1471–1478, Apr. 2021.
- [14] H. Yu, X. Liang, J. Han, and Y. Fang, "Adaptive trajectory tracking control for the quadrotor aerial transportation system landing a payload onto the mobile platform," *IEEE Trans. Ind. Informat.*, vol. 20, no. 1, pp. 23–37, Jan. 2024.
- [15] P. Antal, T. Péni, and R. Tóth, "Autonomous hook-based grasping and transportation with quadcopters," *IEEE Trans. Control Syst. Technol.*, vol. 33, no. 3, pp. 980–990, Mar. 2025.
- [16] P. Antal, T. Péni, and R. Tóth, "Hook-based aerial payload grasping from a moving platform," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2025, pp. 5209–5215.
- [17] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *Proc. IEEE Int. Conf. Intell. Robots Syst.*, 2012, pp. 5026–5033.
- [18] A. Zanelli, J. Frey, F. Messerer, and M. Diehl, "Zero-order robust nonlinear model predictive control with ellipsoidal uncertainty sets," *IFAC-PapersOnLine*, vol. 54, no. 6, pp. 50–57, 2021.
- [19] J. Frey, Y. Gao, F. Messerer, A. Lahr, M. Zeilinger, and M. Diehl, "Efficient zero-order robust optimization for real-time model predictive control with acados," in *Proc. Eur. Control Conf.*, 2024, pp. 3470–3475.
- [20] G. L. Plett, "Lecture notes in applied Kalman filtering," 2018. [Online]. Available: <http://mocha-java.uccs.edu/ECE5550/>
- [21] M. O'Kelly, H. Zheng, D. Karthik, and R. Mangharam, "F1TENTH: An open-source evaluation environment for continuous control and reinforcement learning," in *Proc. NeurIPS Competition Demonstration Track*, 2019, vol. 123, pp. 77–89.
- [22] R. Verschuere et al., "acados—A modular open-source framework for fast embedded optimal control," *Math. Program. Comput.*, vol. 14, no. 1, pp. 147–183, 2022.
- [23] M. Balandat et al., "BoTorch: A framework for efficient monte-carlo Bayesian optimization," in *Proc. Neural Inf. Process. Syst.*, 2020, pp. 21524–21538.