2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

Autonomous Block Assembly for Boom Cranes with Passive Joint Dynamics: Integrated Vision MPC Control

Gerald Ebmer[1], Minh Nhat Vu[1], Tobias Glück[2] and Wolfgang Kemmetmüller[1]

***Abstract*— This paper presents an autonomous control framework for articulated boom cranes performing prefabricated block assembly in construction environments. The key challenge addressed is precise placement control under passive joint dynamics that cause pendulum-like sway, complicating the accurate positioning of building components. Our integrated approach combines real-time vision-based pose estimation of building blocks, collision-aware B-spline path planning, and nonlinear model predictive control (NMPC) to achieve autonomous pickup, placement, and obstacle-avoidance assembly operations. The framework is validated on a laboratory-scale testbed that emulates crane kinematics and passive dynamics while enabling rapid experimentation. The collision-aware planner generates feasible B-spline references in real-time on CPU hardware with anytime performance, while the NMPC controller actively suppresses passive joint sway and tracks the planned trajectory under continuous vision feedback. Experimental results demonstrate autonomous block stacking and obstacle-avoidance assembly, with sway damping reducing settling times by more than an order of magnitude compared to uncontrolled passive dynamics, confirming the real-time feasibility of the integrated approach for construction automation.**

I. Introduction

The assembly of prefabricated building components is a critical bottleneck in modern construction, where heavy elements must be positioned with millimeter-level accuracy under time pressure. Articulated boom cranes are the dominant machinery for such tasks due to their large workspace and payload capacity. However, their under-actuated design—featuring passive joints that allow end-tool self-alignment while reducing actuator requirements—introduces pendulum-like sway dynamics that severely complicate precise placement operations [1], [2].

This sway behavior is particularly problematic for autonomous construction scenarios, where human operators can no longer compensate through intuition and experience. Unlike structured manufacturing environments, construction sites present dynamic, cluttered conditions with irregular obstacles and continuously changing layouts [3], [4], [5]. Robust autonomy, therefore, requires three capabilities in combination: (i) real-time vision feedback to estimate the pose of building components, (ii) collision-free motion planning in cluttered environments, and (iii) active sway damping under passive joint dynamics.

[1]Automation & Control Institute (ACIN), TU Wien, Gusshausstrasse 27-29, 1040 Wien, Austria <ebmer,vu,kemmetmueller>@acin.tuwien.ac.at
[2]Center for Vision, Automation & Control, AIT Austrian Institute of Technology GmbH, Giefinggasse 4, 1210 Vienna, Austria tobias.glueck@ait.ac.at

Fig. 1: Motivation: articulated boom crane with concrete block (left) and developed laboratory-scale setup (right) as a testbed for autonomous assembly.

Existing research has advanced each aspect individually. Vision-based feedback has been applied in crane operations, for instance, using multi-camera systems for simultaneous payload damping and parameter estimation [6], while forestry cranes have integrated stereo vision and grasp planning to achieve autonomous log loading with high success rates [7]. Collision-aware trajectory optimization methods have been proposed for construction cranes, achieving near time-optimal motions under hydraulic constraints in simulation [8]. For sway suppression, both linear and nonlinear MPC schemes have demonstrated substantial reductions in oscillations [9], [10], and dynamic programming approaches achieved up to 90% sway reduction in simulation [11]. Yet these advances remain fragmented: perception, planning, and control are typically addressed in isolation, and no integrated framework exists that unifies them in a closed loop for autonomous prefabricated assembly with articulated boom cranes.

To address this gap, we present a laboratory-scale framework that integrates real-time vision, collision-aware B-spline path generation, and nonlinear model predictive control (NMPC) into a single closed loop. The testbed emulates articulated crane kinematics and passive-joint dynamics while enabling precise performance measurement and rapid experimental cycles that would be infeasible at full scale. Figure 1 illustrates the motivating crane scenario and the developed lab setup. A video of the experiments is available at https://www.acin.tuwien.ac.at/42d7/.

A. Contributions

The main contribution of this work is a unified framework and laboratory testbed for developing autonomous control of

large-scale, under-actuated articulated boom cranes. Specifically:

- **Integrated perception–planning–control architecture:** Real-time vision feedback, collision-free path generation, and nonlinear MPC are combined in a single closed loop that accounts for passive joint dynamics and provides active sway damping.
- **Laboratory-scale validation:** A physical testbed emulating articulated crane dynamics is used to demonstrate autonomous pick-and-place and obstacle-avoidance assembly. The experiments confirm the framework's real-time feasibility and ability to suppress passive-joint sway by more than an order of magnitude, establishing a foundation for transfer to full-scale machinery.

B. Paper's Structure

Section II presents the dynamic model of the lab setup and Section III discusses the nonlinear MPC formulation. Section IV presents the path planner, followed in Section V by the vision-based pose estimation. Section VI details the lab setup and framework integration, followed by the results in Section VII and the conclusion in Section VIII.

II. Modeling

The considered 9-DoF system is depicted on the left side of Figure 2. It consists of the lightweight industrial robot KUKA LBR iiwa R820 with seven actuated DoFs, a cardan joint (passive joint) comprised of two non-actuated DoFs, and the Robotiq 2F-85 gripper. The design of the non-actuated joints resembles a forestry crane [8] equipped with a passive joint.

The equations of motion of the 9-DoF system with the generalized coordinates $\mathbf{q}^{\mathrm{T}} = [q_1, \ldots, q_9]$ can be written as

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) + \mathbf{D}\dot{\mathbf{q}} = \boldsymbol{\tau} \tag{1}$$

with the mass matrix $\mathbf{M}(\mathbf{q})$, the Coriolis matrix $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$, the gravitational forces $\mathbf{g}(\mathbf{q})$, the viscous friction matrix $\mathbf{D}$, and the generalized forces $\boldsymbol{\tau}$ [12], [13]. For further consideration, the nine DoFs are split up into actuated $\mathbf{q}_{\mathrm{a}}^{\mathrm{T}} = [q_1, \ldots, q_7]$ and non-actuated $\mathbf{q}_{\mathrm{u}}^{\mathrm{T}} = [q_8, q_9]$ DoFs. Based on this separation, (1) can be rearranged as

$$\begin{bmatrix} \mathbf{M}_{\mathrm{a}} & \mathbf{M}_{\mathrm{au}} \\ \mathbf{M}_{\mathrm{ua}} & \mathbf{M}_{\mathrm{u}} \end{bmatrix} \begin{bmatrix} \ddot{\mathbf{q}}_{\mathrm{a}} \\ \ddot{\mathbf{q}}_{\mathrm{u}} \end{bmatrix} + \begin{bmatrix} \mathbf{C}_{\mathrm{a}} & \mathbf{C}_{\mathrm{au}} \\ \mathbf{C}_{\mathrm{ua}} & \mathbf{C}_{\mathrm{u}} \end{bmatrix} \begin{bmatrix} \dot{\mathbf{q}}_{\mathrm{a}} \\ \dot{\mathbf{q}}_{\mathrm{u}} \end{bmatrix} + \begin{bmatrix} \mathbf{g}_{\mathrm{a}} \\ \mathbf{g}_{\mathrm{u}} \end{bmatrix} + \begin{bmatrix} \mathbf{D}_{\mathrm{a}} & \mathbf{0} \\ \mathbf{0} & \mathbf{D}_{\mathrm{u}} \end{bmatrix} \begin{bmatrix} \dot{\mathbf{q}}_{\mathrm{a}} \\ \dot{\mathbf{q}}_{\mathrm{u}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\tau}_{\mathrm{a}} \\ \mathbf{0} \end{bmatrix} . \tag{2}$$

Considering the bottom line of (2), the dynamics of the non-actuated system can be represented as

$$\ddot{\mathbf{q}}_{\mathrm{u}} = \mathbf{M}_{\mathrm{u}}^{-1}(-\mathbf{M}_{\mathrm{ua}}\ddot{\mathbf{q}}_{\mathrm{a}} - \mathbf{C}_{\mathrm{ua}}\dot{\mathbf{q}}_{\mathrm{a}} - \mathbf{C}_{\mathrm{u}}\dot{\mathbf{q}}_{\mathrm{u}} - \mathbf{g}_{\mathrm{u}} - \mathbf{D}_{\mathrm{u}}\dot{\mathbf{q}}_{\mathrm{u}}) . \tag{3}$$

To reduce the complexity of the model (3), the Coriolis terms $\mathbf{C}_{\mathrm{ua}}\dot{\mathbf{q}}_{\mathrm{a}}$ and $\mathbf{C}_{\mathrm{u}}\dot{\mathbf{q}}_{\mathrm{u}}$ are neglected as their contribution to the system dynamics is minor; see e. g. [14]. This yields the simplified dynamics of the non-actuated system

$$\ddot{\mathbf{q}}_{\mathrm{u}} = \mathbf{M}_{\mathrm{u}}^{-1}(-\mathbf{M}_{\mathrm{ua}}\ddot{\mathbf{q}}_{\mathrm{a}} - \mathbf{g}_{\mathrm{u}} - \mathbf{D}_{\mathrm{u}}\dot{\mathbf{q}}_{\mathrm{u}}) . \tag{4}$$

It is further assumed that the actuated joints are controlled by a sub-ordinate controller such that the system input $\mathbf{u}$ for the nonlinear MPC can be chosen as $\mathbf{u} = \ddot{\mathbf{q}}_{\mathrm{a}}$. The resulting system dynamics are

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}) = \begin{bmatrix} \boldsymbol{\omega}_{\mathrm{a}} \\ \boldsymbol{\omega}_{\mathrm{u}} \\ \mathbf{u} \\ \mathbf{M}_{\mathrm{u}}^{-1}(-\mathbf{M}_{\mathrm{ua}}\mathbf{u} - \mathbf{g}_{\mathrm{u}} - \mathbf{D}_{\mathrm{u}}\boldsymbol{\omega}_{\mathrm{u}}) \end{bmatrix} , \tag{5}$$

with the system state $\mathbf{x}^{\mathrm{T}} = [\mathbf{q}^{\mathrm{T}}, \boldsymbol{\omega}^{\mathrm{T}}]$ and $\boldsymbol{\omega} = \dot{\mathbf{q}}$.

III. Path-Following MPC Incorporating Passive Joint Dynamics

The path-following MPC approach is formulated as a finite-horizon optimal control problem that optimizes the path parameter progression and the system's motion along a predefined spline-based path. Due to the under-actuated nature of the system, only the gripper's position and yaw angle can be directly controlled, necessitating a 4D task space representation $\boldsymbol{\xi} = \left[x, y, z, \psi\right]^{\mathrm{T}}$ where the passive joint dynamics determine roll and pitch orientations. The MPC formulation incorporates the passive joint dynamics while advancing the path to enable precise assembly operations.

A. Spline-Based Path Parameterization

Task space path parameterization is preferred for assembly operations as it provides smoother end-effector trajectories than joint space paths, which introduce irregular Cartesian motions due to kinematic nonlinearities. The path parameter $s \in [0, 1]$ represents progression along the 4D path defined by a B-spline $\boldsymbol{\xi}_{\mathrm{d}}(s) = \sum_{i=0}^{n_{\mathrm{cp}}-1} \mathbf{c}_i B_{i,k}(s)$ from a start pose $\boldsymbol{\xi}_{\mathrm{s}} = \boldsymbol{\xi}_{\mathrm{d}}(0)$ to an end pose $\boldsymbol{\xi}_{\mathrm{e}} = \boldsymbol{\xi}_{\mathrm{d}}(1)$, where $B_{i,k}(s)$ are B-spline basis functions of degree k, and each control point $\mathbf{c}_i = \left[x_i, y_i, z_i, \psi_i\right]^{\mathrm{T}}$, $i = 0, \ldots, n_{\mathrm{cp}} - 1$ contains position and yaw information.

B. Inverse Kinematics and Steady State of the Under-actuated System

Due to the under-actuated dynamics, inverse kinematics solutions for a stationary end pose $\boldsymbol{\xi}_{\mathrm{e}} = \left[\mathbf{p}_{\mathrm{e}}^{\mathrm{T}}, \psi_{\mathrm{e}}\right]^{\mathrm{T}}$ must correspond to a steady state of the non-actuated subsystem to avoid unwanted sway motions. At steady state, both accelerations $\ddot{\mathbf{q}}_{\mathrm{u}}$ and velocities $\dot{\mathbf{q}}_{\mathrm{u}}$ of the under-actuated joints are zero. From (4), the steady-state condition requires $\mathbf{g}_{\mathrm{u}}(\mathbf{q}) = \mathbf{0}$, where $\mathbf{g}_{\mathrm{u}}(\mathbf{q})$ represents the gravitational forces acting on the under-actuated joints. Using the error formulation similar to [15], the steady-state configuration $\mathbf{q}_{\mathrm{e}}$ is obtained by solving

$$\mathbf{q}_{\mathrm{e}} = \arg\min_{\mathbf{q}} \left[\left\| \mathbf{p}(\mathbf{q}) - \mathbf{p}_{\mathrm{e}} \right\|_2^2 + \left\| \mathbf{R}(\psi_{\mathrm{e}})^{\mathrm{T}} \mathbf{R}(\mathbf{q}) - \mathbf{I} \right\|_{\mathrm{F}}^2 \right] \tag{6}$$

$$\text{subject to} \quad \mathbf{g}_{\mathrm{u}}(\mathbf{q}) = \mathbf{0}$$

where $\|\cdot\|_{\mathrm{F}}$ denotes the Frobenius norm. The optimization problem (6) is solved with Ipopt [16].

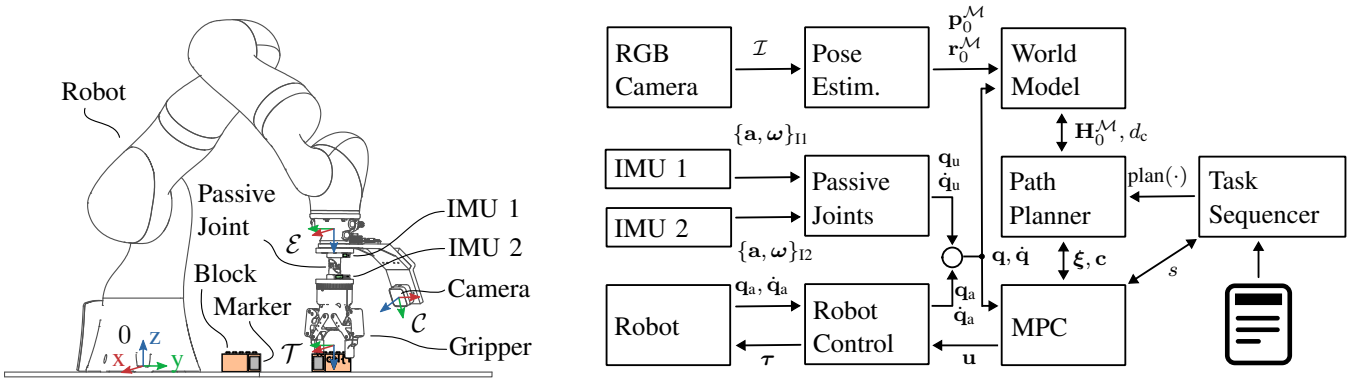


Fig. 2: System overview. Left: schematic of the laboratory setup with robot, passive joint, gripper, blocks, and reference frames 0 (base), $\mathcal{E}$ (end-effector), $\mathcal{T}$ (tool center), and $\mathcal{C}$ (camera). Right: block diagram of the integrated framework, illustrating the interconnection of perception, path planning, MPC control, and the environment model. The task sequencer executes pickup and placement routines based on a configuration file specifying the schedule.

C. Optimal Control Problem Formulation

Our control strategy aims are twofold: Firstly, it should incorporate the under-actuated system dynamics while stabilizing the gripper, and secondly, it should progress the path. For this we augment the original system state $\mathbf{x}$ with the path parameter s and its velocity $\dot{s}$ resulting in the augmented system state $\bar{\mathbf{x}} = \left[\mathbf{x}^{\mathrm{T}}, s, \dot{s}\right]^{\mathrm{T}} \in \mathbb{R}^{2n_q+2}$, as well as the augmented system input comprising the acceleration of the seven actuated joints and the path parameter's second time derivative as input $\bar{\mathbf{u}} = \left[\mathbf{u}^{\mathrm{T}}, v\right]^{\mathrm{T}} = \left[\ddot{\mathbf{q}}_{\mathrm{a}}^{\mathrm{T}}, \ddot{s}\right]^{\mathrm{T}} \in \mathbb{R}^{n_{\mathrm{a}}+1}$

Based on (5), the augmented system dynamics result in

$$\dot{\bar{\mathbf{x}}} = \bar{\mathbf{f}}(\bar{\mathbf{x}}, \bar{\mathbf{u}}) = \begin{bmatrix} \mathbf{f}(\mathbf{x}, \mathbf{u}) \\ \dot{s} \\ v \end{bmatrix}. \tag{7}$$

The optimal control problem for the path-following MPC is defined as

$$\min_{\substack{\bar{\mathbf{x}}_{1|n},\dots,\bar{\mathbf{x}}_{N|n} \\ \bar{\mathbf{u}}_{1|n},\dots,\bar{\mathbf{u}}_{N-1|n}}} \sum_{k=0}^{N-1} l\big(\bar{\mathbf{x}}_{k|n}, \bar{\mathbf{u}}_{k|n}\big) + m\big(\bar{\mathbf{x}}_{N|n}\big) \tag{8a}$$

$$\text{subject to} \quad \bar{\mathbf{x}}_{k+1|n} = \mathbf{F}(\bar{\mathbf{x}}_{k|n}, \bar{\mathbf{u}}_{k|n}) \tag{8b}$$

$$\bar{\mathbf{x}}_{0|n} = \bar{\mathbf{x}}(t_n) \tag{8c}$$

$$\bar{\mathbf{x}}^{-} \leq \bar{\mathbf{x}}_{k|n} \leq \bar{\mathbf{x}}^{+}, \quad \bar{\mathbf{u}}^{-} \leq \bar{\mathbf{u}}_{k|n} \leq \bar{\mathbf{u}}^{+} \tag{8d}$$

where n indicates the iteration of the OCP and $k = 0, \dots, N$ refers to the collocation points over the prediction horizon with length N. The discrete-time system dynamics $\bar{\mathbf{x}}_{k+1|n} = \mathbf{F}(\cdot)$ are obtained through implicit Runge-Kutta integration of (7) with the discretization timestep T_{h}. The state limits $\bar{\mathbf{x}}^{\pm} = \left[\mathbf{q}_{\mathrm{a}}^{\pm}, \mathbf{q}_{\mathrm{u}}^{\pm}, \dot{\mathbf{q}}_{\mathrm{a}}^{\pm}, \dot{\mathbf{q}}_{\mathrm{u}}^{\pm}, s^{\pm}, \dot{s}^{\pm}\right]^{\mathrm{T}}$ and input limits $\bar{\mathbf{u}}^{\pm} = \left[\mathbf{u}^{\pm}, v^{\pm}\right]^{\mathrm{T}}$ in (8d) ensure the physical feasibility of the motion by including the joint limits and velocity limits for the robot $\mathbf{q}_{\mathrm{a}}^{\pm} = \pm\left[162, 114, 162, 114, 162, 114, 166\right]^{\mathrm{T}}$ in °, $\mathbf{q}_{\mathrm{u}}^{\pm} = \pm\left[43, 43\right]^{\mathrm{T}}$ in ° and $\dot{\mathbf{q}}_{\mathrm{a}}^{\pm} = \pm\left[85, 85, 100, 75, 130, 135, 135\right]^{\mathrm{T}}$ in °/s. The remaining parameters were chosen empirically to achieve good performance in the experiments reported in Section VII $\dot{\mathbf{q}}_{\mathrm{u}}^{\pm} = \pm\left[50, 50\right]^{\mathrm{T}}$ in °/s, $s^{+} = 1$, $s^{-} = 0$, $\dot{s}^{\pm} = \pm 1$ in $1/\mathrm{s}$ and $\mathbf{u}^{\pm} = \pm\mathbf{1}$ in $\mathrm{rad/s^2}$, $v^{\pm} = \pm 1$ in $1/\mathrm{s^2}$, where $\mathbf{1}$ denotes an all-ones vector of appropriate dimension.

The Lagrange cost $l\big(\bar{\mathbf{x}}_{k|n}, \bar{\mathbf{u}}_{k|n}\big)$ of the cost function (8a) consists of task space tracking, joint space regularization, and control effort minimization part in the form of

$$\begin{aligned} l(\bar{\mathbf{x}}_{k|n}, \bar{\mathbf{u}}_{k|n}) = {} & \|\tilde{\mathbf{p}}(\mathbf{q}_{k|n})\|^2_{\mathbf{Q}_{\mathrm{pos}}} + \|\tilde{\psi}(\mathbf{q}_{k|n})\|^2_{Q_{\mathrm{rot}}} \\ & + \|\tilde{\mathbf{q}}_{k|n}\|^2_{\mathbf{Q}_q} + \|\dot{\tilde{\mathbf{q}}}_{k|n}\|^2_{\mathbf{Q}_{\dot{q}}} \\ & + \|\tilde{s}_{k|n}\|^2_{\eta Q_s} + \|\dot{\tilde{s}}_{k|n}\|^2_{Q_{\dot{s}}} \\ & + \|\mathbf{u}_{k|n}\|^2_{\mathbf{R}_u} + \|v_{k|n}\|^2_{R_v} \end{aligned} \tag{9}$$

with weighted norms $\|\cdot\|_Q$. The task space errors $\tilde{\mathbf{p}}(\mathbf{q}_{k|n}) = \mathbf{p}(\mathbf{q}_{k|n}) - \mathbf{p}_{\mathrm{d}}(s_{k|n})$ and $\tilde{\psi}(\mathbf{q}_{k|n}) = \psi(\mathbf{q}_{k|n}) - \psi_{\mathrm{d}}(s_{k|n})$ with the Cartesian position $\mathbf{p}(\mathbf{q}_{k|n})$ and the gripper's yaw angle $\psi(\mathbf{q}_{k|n})$ depend on the joint configuration through forward kinematics, while $\mathbf{p}_{\mathrm{d}}(s_{k|n})$, and $\psi_{\mathrm{d}}(s_{k|n})$ are obtained from the evaluation of the reference path parameterized as B-spline. The joint-space regularization terms are defined relative to a nominal posture $\mathbf{q}_{\mathrm{nom}}$ (typically $\mathbf{q}_{\mathrm{nom}} = \mathbf{q}_{\mathrm{e}}$) and zero velocity as $\tilde{\mathbf{q}}_{k|n} = \mathbf{q}_{k|n} - \mathbf{q}_{\mathrm{nom}}$, $\dot{\tilde{\mathbf{q}}}_{k|n} = \dot{\mathbf{q}}_{k|n}$. The path-parameter tracking error is $\tilde{s}_{k|n} = s_{k|n} - s_{\mathrm{d}}$ with regularization $\dot{\tilde{s}}_{k|n} = \dot{s}_{k|n}$. The path parameter weight Q_s is scaled by $\eta = 1/\max(\|\mathbf{p}_{\mathrm{e}} - \mathbf{p}_{\mathrm{s}}\|_2, 0.1)$ where $\mathbf{p}_{\mathrm{s}}$ and $\mathbf{p}_{\mathrm{e}}$ denote the path's start and end position, respectively.

The Mayer cost term at the final stage $m\big(\bar{\mathbf{x}}_{N|n}\big)$ is similar to the Lagrange cost (9), except that it excludes $\|\mathbf{u}_{k|n}\|^2_{\mathbf{R}_u}$ and $\|v_{k|n}\|^2_{R_v}$. The values of the weight matrices and MPC parameters are listed in Section VII.

Kinematic redundancy is resolved by regularizing joints toward a nominal posture $\mathbf{q}_{\mathrm{nom}}$ with large weights in $\mathbf{Q}_q$ assigned to joints that do not exist in articulated boom cranes (e. g., the robot's axial rotations q_2 and q_6), while $\mathbf{Q}_{\dot{q}}$ damps joint velocities.

D. Real-Time Implementation

The computational efficiency required for real-time control is achieved through several key implementation choices. Traditional Euler-Lagrange formulations to obtain the equations of motion result in large symbolic expressions that lead to prohibitively long compile times and, most importantly, high computational cost, often producing hundreds

of megabytes of generated code, particularly for computing the computationally expensive Hessian matrix. To address these challenges, the implementation utilizes the Recursive Newton-Euler Algorithm (RNEA) [17] as implemented in the Pinocchio library [18]. This approach, combined with CasADi's automatic differentiation capabilities [19], enables efficient computation of the robot dynamics and their derivatives. The seamless integration of CasADi with the ACADOS optimal control framework [20] allows for automatic generation, compilation, and execution of optimized solver binaries. The B-spline path parameterization used in (9) is implemented using a custom-made CasADi-compatible library, ensuring efficient evaluation of reference trajectories and their derivatives within the optimization framework.

The ACADOS-generated solver employs the Real-Time Iteration (RTI) algorithm [21] optimized for real-time performance, using a single QP iteration per MPC cycle with HPIPM as the QP solver featuring partial condensing and a maximum of 20 internal iterations [22]. The configuration includes warm starting for accelerated convergence, merit function backtracking for robust convergence, exact Hessian computation, and implicit Runge-Kutta (IRK) integration for numerical stability. The computation times of (8) with the given framework are discussed in Section VII.

IV. Collision-Aware Path Planning

Path-following MPC requires task-space references that are feasible with respect to obstacles and can be updated at a fast control rate. We employ a lightweight stochastic ensemble method for quadratic B-splines in the 4-D task space (x, y, z, ψ). The approach is inspired by spline-based motion planning methods [23], [24] and stochastic sampling strategies related to MPPI and Cross-Entropy methods [25], [26]. In contrast to prior GPU-based implementations [27], the proposed method is realized as a CPU-optimized implementation with OpenMP parallelization, targeting deployment in real-time control systems and achieving per-cycle runtimes of only a few milliseconds.

The planner operates in an iterative manner: Per planner iteration, an ensemble of path candidates is sampled from a distribution, the best candidate is selected, and the distribution is updated. A single planner iteration does not need to converge; instead, quality improves across successive cycles by warm-starting from the previous distribution and best path. We use "anytime" in this specific sense: *bounded computation per planner iteration, with improving solutions across iterations.*

A. Problem Formulation

Let $\boldsymbol{\xi}(s; \mathbf{c}) \in \mathbb{R}^4$ denote a quadratic B-spline curve with clamped knots and control points $\mathbf{c}$ (for brevity, write $\boldsymbol{\xi}(s) := \boldsymbol{\xi}(s; \mathbf{c})$). Start and goal are fixed, $\boldsymbol{\xi}(0) = \boldsymbol{\xi}_\mathrm{s}$, $\boldsymbol{\xi}(1) = \boldsymbol{\xi}_\mathrm{e}$, while interior via points $\{\boldsymbol{\xi}_{\mathrm{v},i}\}_{i=1}^{K}$ adapt to avoid collisions and shorten the path. The control points are reconstructed by spline interpolation through the start, interior, and goal points.

B. Ensemble Sampling and Distribution Update

In each planner iteration, an ensemble of candidate paths $\mathcal{P} = \{\boldsymbol{\xi}^{(m)}\}_{m=0}^{M}$ from $\boldsymbol{\xi}_\mathrm{s}$ to $\boldsymbol{\xi}_\mathrm{e}$ is generated by sampling interior via points from the current distribution,

$$\boldsymbol{\xi}_{\mathrm{v},i}^{(m)} \sim \mathcal{N}(\boldsymbol{\mu}_i,\ \boldsymbol{\Sigma}_i), \qquad i = 1, \ldots, K, \tag{10}$$

with the following constraints: (x, y, z) are truncated to workspace bounds, z is projected above $z_{\min}$, and yaw ψ is wrapped into the valid angular range. Together with the fixed start $\boldsymbol{\xi}_{\mathrm{v},0} = \boldsymbol{\xi}_\mathrm{s}$ and goal $\boldsymbol{\xi}_{\mathrm{v},K+1} = \boldsymbol{\xi}_\mathrm{e}$, these points define a candidate spline $\boldsymbol{\xi}^{(m)}$. For the initial planner iteration (cold start) the mean value $\boldsymbol{\mu}_0$ is computed as linear interpolation from $\boldsymbol{\xi}_\mathrm{s}$ to $\boldsymbol{\xi}_\mathrm{e}$ and $\boldsymbol{\Sigma}_i = \sigma_{\text{init}}^2 \mathbf{I}$ with the parameter σ_{init}.

Each candidate $\boldsymbol{\xi}^{(m)}$ is discretized at N_c equidistant samples $\{s_j\}$. At each path pose $\boldsymbol{\xi}^{(m)}(s_j) = \left[(\mathbf{p}_j^{(m)})^\mathrm{T}, \psi_j^{(m)}\right]^\mathrm{T}$ we compute: arc length increment $\left\|\mathbf{p}_j^{(m)} - \mathbf{p}_{j-1}^{(m)}\right\|$, collision cost $d(\boldsymbol{\xi}_j^{(m)})$, and a floor penalty

$$P_{\text{floor}}(\mathbf{p}_j^{(m)}) = \begin{cases} \alpha\Big(z_{\min} + \delta - z_j^{(m)}\Big)^2, & z_j^{(m)} < z_{\min} + \delta, \\ 0 & \text{otherwise.} \end{cases}$$

The composite cost is

$$J(\boldsymbol{\xi}^{(m)}) = \sum_{j=1}^{N_c - 1} \left\|\mathbf{p}_j^{(m)} - \mathbf{p}_{j-1}^{(m)}\right\| + w_\mathrm{c}\big(d(\boldsymbol{\xi}_j^{(m)}) + P_{\text{floor}}(\mathbf{p}_j^{(m)})\big)\ , \tag{11}$$

with the factor $w_\mathrm{c} > 0$. The collision cost $d(\boldsymbol{\xi}_j^{(m)})$ is computed with Mujoco's GJK/EPA collision pipeline [28].

After evaluation, elites are selected by cost J with normalized log-weights

$$w_m \ = \ \frac{\ln\big(M + 0.5\big) \ - \ \ln(m)}{\sum_{\nu=1}^{M}\Big[\ln\big(M + 0.5\big) \ - \ \ln(\nu)\Big]}\ .$$

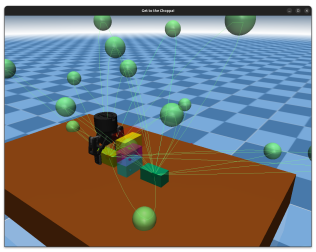
and for each via index i, the mean and variance are updated (warm start) as

$$\begin{aligned} \boldsymbol{\mu}_i &\leftarrow \sum_{m \in \mathcal{E}} w_m \boldsymbol{\xi}_{\mathrm{v},i}^{(m)}, \\ \boldsymbol{\Sigma}_i &\leftarrow \sum_{m \in \mathcal{E}} w_m (\boldsymbol{\xi}_{\mathrm{v},i}^{(m)} - \boldsymbol{\mu}_i)(\boldsymbol{\xi}_{\mathrm{v},i}^{(m)} - \boldsymbol{\mu}_i)^\mathrm{T} + \epsilon \mathbf{I} \end{aligned} \tag{12}$$

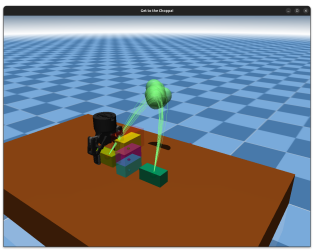
where $\epsilon \mathbf{I}$ provides numerical stability. Exploration is adapted multiplicatively: if collision-free candidates exist, $\boldsymbol{\Sigma}_i$ contracts; otherwise, it expands. The reference path for MPC is the feasible candidate with minimal J.

C. Benchmark Results

The planner was benchmarked on the scenario depicted in Figure 3 (4-DoF, one interior via point) over N=100 runs. We evaluate the planner with different target runtimes. Each run is stopped after the target time has elapsed; however, we allow the current iteration to finish before reporting. Thus, the reported times may slightly exceed the nominal target. Table I reports results for target runtimes $10, 20, 50$ ms.



(a) Initial ensemble sampled around the straight-line initialization.



(b) Ensemble after 10 refinement iterations.

Fig. 3: Evolution of the stochastic ensemble (15 candidates) with an obstacle wall of three stacked blocks. Candidates start with wide variability (a) and contract toward a feasible collision-free region (b).

The first cycle is a cold start from the linear initialization; subsequent cycles continue with a warm-started distribution and the best path. Even under tight runtimes, the planner reliably produces feasible references, while larger runtimes yield shorter paths and higher iteration counts.

TABLE I: Planner benchmark results (N=100 runs). Cold: first cycle from linear initialization; Warm: continued refinement with warm start. Runtime is per cycle. All runs use ensemble size M=15, one interior via point (K=1), initial standard deviation σ_{init}=0.2, and N_c=40 collision checks per path.

Target	Mode	Success	Time	Iter.	Length
10 ms	Cold	95%	(12.6 ± 2.9) ms	2.00	0.780 m
	Warm	94%	(12.5 ± 3.4) ms	2.23	0.753 m
20 ms	Cold	98%	(23.1 ± 2.4) ms	3.99	0.685 m
	Warm	100%	(22.7 ± 2.7) ms	5.41	0.598 m
50 ms	Cold	100%	(53.6 ± 3.9) ms	13.45	0.497 m
	Warm	100%	(53.3 ± 3.7) ms	12.66	0.521 m

V. Pose Estimation

Using fiducial markers, our vision pipeline estimates the pose of blocks from RGB images $\mathcal{I}$. Although the generality of marker-based approaches is limited, they provide a simple and computationally efficient solution suitable for controlled laboratory validation under real-time constraints.

Given a calibrated camera image $\mathcal{I}$ with intrinsics $\mathbf{K}$ and distortion parameters, we detect ArUco markers $\mathcal{M}$ [29] and estimate their pose $\mathbf{H}_{\mathcal{C}}^{\mathcal{M}}$ (homogeneous transformation) using the IPPE PnP algorithm [30]. With the fixed hand–eye calibration $\mathbf{H}_{\mathcal{T}}^{\mathcal{C}}$ and the robot forward kinematics $\mathbf{H}_0^{\mathcal{T}}(\mathbf{q})$, the marker pose in the robot base frame is obtained as

$$\mathbf{H}_0^{\mathcal{M}} = \mathbf{H}_0^{\mathcal{T}}(\mathbf{q})\mathbf{H}_{\mathcal{T}}^{\mathcal{C}}\mathbf{H}_{\mathcal{C}}^{\mathcal{M}} \Rightarrow \left(\mathbf{p}_0^{\mathcal{M}}, \mathbf{r}_0^{\mathcal{M}}\right), \tag{13}$$

with position $\mathbf{p}_0^{\mathcal{M}} \in \mathbb{R}^3$ and orientation $\mathbf{r}_0^{\mathcal{M}} \in \mathbb{H}$ (unit quaternion).

Since the block motion is unknown, we adopt a constant-velocity motion prior as a generic assumption. The position $\mathbf{p}_0^{\mathcal{M}}$ is filtered with a Kalman filter on the state $\hat{\mathbf{x}}_{p,k} = \left[\mathbf{p}_{0,k}^{\mathcal{M}}, \mathbf{v}_{0,k}^{\mathcal{M}}\right]^{\mathrm{T}}$, while the orientation $\mathbf{r}_0^{\mathcal{M}}$ is smoothed using a multiplicative EKF (MEKF) on $\mathrm{SO}(3)$ with quaternion error representation [31], [32]. Both filters run at the camera rate using the measured frame interval Δt_k. The filtered 6D poses are fed into the world model (see Fig. 2) and serve as goal references $\boldsymbol{\xi}_{\mathrm{e}}$ for the path planner, which updates the path executed by the MPC.

VI. Laboratory Setup

The experimental platform comprises a KUKA LBR iiwa 14 R820 equipped with a Robotiq 2F-85 parallel-jaw gripper and two passive joint DoFs. The unactuated coordinates q_8 and q_9 correspond to the relative roll and pitch between the robot end-effector and the gripper frame. These angles are inferred from a pair of IMUs (ADIS16460, Analog Devices) rigidly mounted on the end-effector and the gripper. Raw angular velocities are first debiased; orientation estimates are then obtained using a Madgwick filter [33]. Let $\mathbf{R}_{\mathcal{W}}^{\mathrm{I1}}$ and $\mathbf{R}_{\mathcal{W}}^{\mathrm{I2}}$ denote the orientation (rotation matrices) of the end-effector-mounted and gripper-mounted IMUs in the inertial frame. The relative rotation is computed as $\mathbf{R}_{\mathrm{I1}}^{\mathrm{I2}} = \left(\mathbf{R}_{\mathcal{W}}^{\mathrm{I1}}\right)^{\mathrm{T}}\mathbf{R}_{\mathcal{W}}^{\mathrm{I2}}$, from which the roll and pitch components are extracted to define q_8 and q_9. The IMU sampling rate is 500 Hz. An Intel RealSense D405 depth camera is rigidly co-located with the robot's end-effector and operated at 15 Hz for close-range perception and ArUco-based pose estimation.

All device drivers for the IMUs and the camera run on a Raspberry Pi 4 (Ubuntu, ROS 2). The Madgwick orientation filter, ArUco pose filtering, task sequencer, path planner, and the model predictive controller (MPC) execute on a workstation (Ubuntu 22.04, Intel Core i7-12700K, 32 GB RAM) running ROS 2 with Eclipse Cyclone DDS as middleware. Hard real-time joint control is deployed on a Beckhoff TwinCAT PC at $T_{\mathrm{c}} = 125\,\mu$s (8 kHz); ROS 2 and TwinCAT communicate via UDP, and TwinCAT interfaces with the robot over EtherCAT.

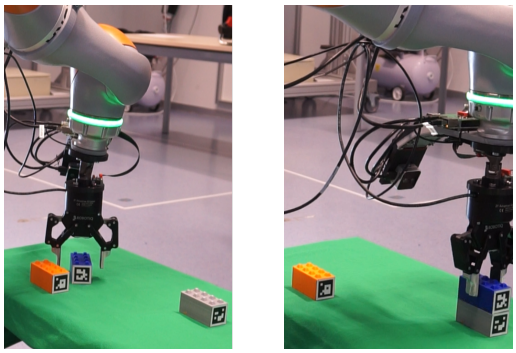
The weights in (9) were tuned empirically in experiments, with high values on position and orientation tracking ($\mathbf{Q}_{\mathrm{pos}}, Q_{\mathrm{rot}}$) to ensure accuracy, and small posture and velocity regularization terms to stabilize redundancy, thereby balancing path-tracking precision, sway damping, and control effort in the experimental setup. The horizon length in (8) is chosen as $N = 25$ steps, and the time discretization of the horizon is $T_{\mathrm{h}} = 30\,\mathrm{ms}$, which is fast enough for the passive joint dynamics and long enough horizon for smooth path-following. The MPC runs with $T_{\mathrm{s}} = 30\,\mathrm{ms}$ and publishes a sequence of joint-accelerations $\mathbf{u}_{i|n} \equiv \ddot{\mathbf{q}}_{\mathrm{a}}(t_{i|n})$ with $t_{i|n} = nT_{\mathrm{s}} + iT_{\mathrm{h}}$ and $i = 0, \ldots, N$. In this work, real-time refers to the property that all MPC computations are completed within the fixed sampling interval T_{s}.

Since the MPC and torque control loops of the robot operate at different cycle times ($T_s = 30\,\mathrm{ms}$ vs. $T_c = 125\,\mu$s), signal consistency requires explicit up- and down-sampling. Linear interpolation of accelerations ensures that the inner torque loop receives continuous, cycle-synchronous references. The robot is controlled by a computed-torque (CT) controller [12] with a sampling period $T_{\mathrm{c}} = 125\,\mu$s. To obtain the reference joint state and velocity at the control rate, the acceleration sequence $\mathbf{u}_{i|n} \equiv \ddot{\mathbf{q}}_{\mathrm{a}}(t_{i|n})$ is first interpolated and then integrated with the semi-implicit Euler method. The parameters for the computed-torque controller

are obtained from (1) with the mass of the passive joint and of the gripper included in the last link.

A lightweight world model maintains the state of movable workpieces, including their estimated 6D poses and assembly status. This world model parametrizes a MuJoCo simulation updated online and queried for collisions during path planning (see Fig. 2). The simulation provides the collision environment that the planner uses, ensuring that planning queries remain consistent with the real robot and workpieces. This way, perception, planning, and control are tightly coupled through a unified scene representation.

VII. Experimental Validation



(a) Pickup. (b) Placement.

Fig. 4: Block pickup and placement. In (a), a randomly placed block is grasped using vision-based pose estimation feedback. In (b), the block is placed on top of an existing block at a predefined goal, consistent with nominal assembly targets.

The framework was validated in three experiments: The first and second experiments illustrate block pickup and placement tasks with online vision updates, and the third experiment shows the sway-damping performance of the presented method. For the first and second experiments, the planner iterations are executed at $1\,\mathrm{s}$ intervals. This interval was chosen to make replanning events clearly visible in the plots. The architecture supports faster updates limited by perception and computational constraints. In all experiments, we use $K = 1$, i. e. we have the start, goal, and one interior via point. This minimal choice is sufficient for collision avoidance in the considered assembly scenarios while keeping the optimization lightweight. Determining the optimal number of via points for more complex tasks is an open question and left for future work. However, it is important to note that the proposed method is suitable for an arbitrary number of via points as it scales linearly with K, see [34].

In the first experiment, depicted in Figure 4, blocks are picked up from random poses and stacked on top of each other. Figure 5 compares the reference trajectory (dashed) with the executed motion (solid), with replanning events marked by vertical dashed lines. The corresponding errors are shown in Fig. 5, remaining below $3\,\mathrm{mm}$ and 3° during slow approach phases, while transient deviations up to $15\,\mathrm{mm}$ and 5° occur during faster motions (e.g. around $t = 11\,\mathrm{s}$). These values remain within the tolerances required for block stacking in the laboratory setup.

The evolution of the path parameter s, its time derivative $\dot{s}$, and v given in Figure 6 illustrates how the path-following MPC advances along the spline under real-time updates. Solver performance is summarized in Fig. 7: QP solve times remain below $20\,\mathrm{ms}$, iteration counts stay moderate, and residuals confirm stable convergence of the RTI scheme. Together, these results demonstrate that the integrated framework achieves real-time feasibility on CPU hardware while maintaining sufficient accuracy for autonomous assembly.

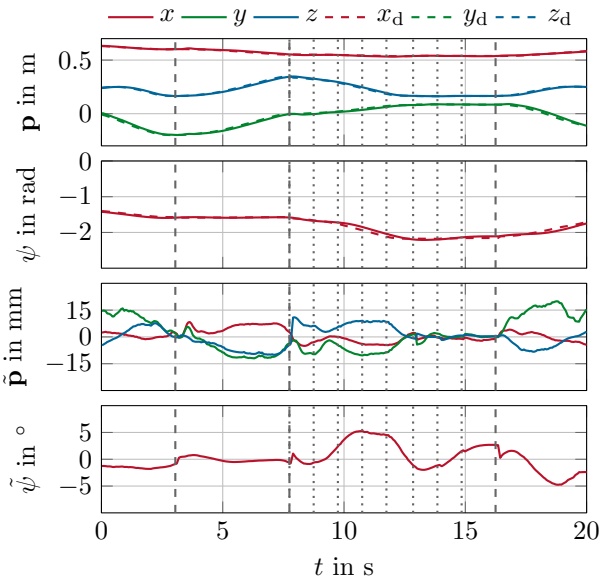


Fig. 5: Reference tracking and error during block pickup and placement (see Figure 4). Vertical dashed lines indicate replanning events (new path initialization), while dotted lines mark vision updates (goal corrections from pose estimation). From top to bottom: Cartesian position $\mathbf{p}$, yaw angle ψ, Cartesian position errors $\tilde{\mathbf{p}}$ and yaw angle error $\tilde{\psi}$.

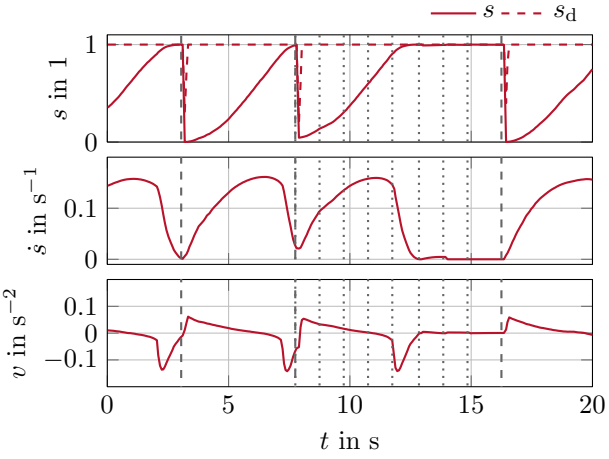


Fig. 6: Path progress during block pickup and placement. Top: path parameter s. Middle: time derivative $\dot{s}$. Bottom: input v.

In the second experiment, blocks are picked up on one side of an obstacle wall and stacked on the opposite side. Figure 8 illustrates the experiment sequence. The accompanying video

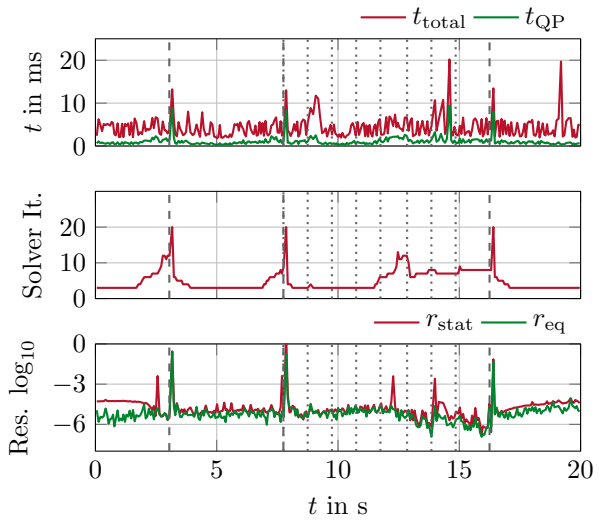


Fig. 7: MPC performance metrics during block pickup and placement. Top: computation times, with t_{total} the overall MPC iteration time and t_{QP} the QP solver time. Middle: number of QP solver iterations per MPC step. Bottom: convergence residuals, where r_{stat} measures first-order optimality (gradient norm) and r_{eq} the satisfaction of system dynamics constraints.

(a) First pick up. (b) Traverse. (c) First placement.

(d) Second pick up. (e) Traverse. (f) Sec. placement.

Fig. 8: Assembly of blocks with an obstacle wall (see Figure 3). The wall consists of three stacked blocks separating pickup and placement locations. Two blocks are transferred across the obstacle in sequence, each consisting of pickup, traverse, and placement.

provides a clearer view of this experiment, particularly the gripper re-adjustments during pickup based on vision feedback.

The third experiment depicted in Figure 9 illustrates the damping of passive joint oscillations achieved by the MPC controller. The gripper was manually perturbed to excite the sway angles q_8 and q_9 to evaluate performance. The controller was active from 0 s to 15 s and then deactivated, exposing the natural dynamics of the under-actuated joints. A damped-sine fit was applied to both cases, and the key parameters are summarized in Table II. The results highlight

TABLE II: Damping characteristics of q_9 with MPC active vs. inactive. With MPC active, the decay rate and damping ratio increase by more than an order of magnitude, reducing the settling time from 17.8 s to below 1 s.

	Decay rate σ	Damping ratio ζ	$T_{90\to10}$
MPC on	2.50/s	0.23	0.88 s
MPC off	0.12/s	0.013	17.8 s

the exceptional damping achieved by the MPC controller: the effective damping ratio increases from $\zeta = 0.013$ (barely damped) to $\zeta = 0.23$, reducing the settling time by a factor of more than 20 (from 17.8 s to less than 0.9 s). Notably, the oscillation frequency remains nearly unchanged (1.67 Hz vs. 1.54 Hz), showing that the improvement is due to active damping rather than altered dynamics. This confirms that the MPC effectively suppresses passive joint sway, stabilizing the gripper motion in real time. All experimental results are summarized in an accompanying video available at `https://www.acin.tuwien.ac.at/42d7/`.

VIII. Conclusion

This paper presents a unified perception–planning–control framework for under-actuated manipulators with passive joints, realized on a laboratory-scale testbed designed to

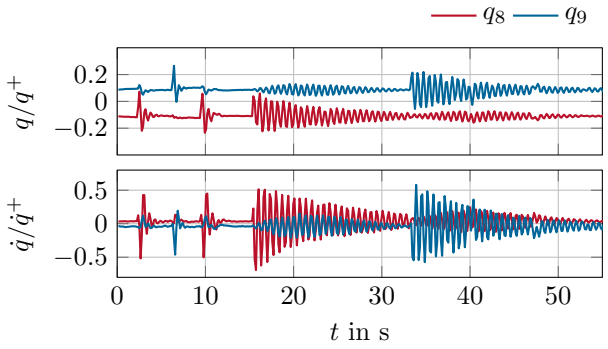


Fig. 9: Damping performance of the MPC controller under external excitation. The controller is active from $t = 0$ s to $t = 15$ s, after which the natural, lightly damped sway motion of the passive joints becomes visible. Quantitative parameters of the fitted oscillations are reported in Table II.

emulate articulated boom cranes. Integrating real-time vision feedback, collision-aware path generation, and nonlinear MPC enables closed-loop execution of assembly tasks under dynamic and cluttered conditions.

Experiments on the testbed demonstrated autonomous block pickup, placement, and obstacle-avoidance assembly with online vision updates. The results confirmed both the real-time feasibility of the architecture and its ability to actively damp passive-joint sway, reducing settling times by more than an order of magnitude.

The laboratory-scale setup proved to be an effective intermediate environment: simple enough for rapid prototyping and controlled experimentation, yet rich enough to capture the key challenges of under-actuated crane dynamics. This

bridge between tabletop robotics and full-scale machinery allows systematic development of autonomy concepts under realistic but manageable conditions.

Future work will extend the vision system beyond fiducial markers, explore richer assembly tasks, and investigate the transition to large-scale machinery.

REFERENCES

[1] E. Kowsari and R. Ghabcheloo, "Optimal sway motion reduction in forestry cranes," *Frontiers in Robotics and AI*, vol. 11, p. 1417741, Aug. 2024.

[2] B. He, S. Wang, and Y. Liu, "Underactuated robotics: a review," *International Journal of Advanced Robotic Systems*, vol. 16, no. 4, p. 1729881419862164, 2019.

[3] B. Xiao, C. Chen, and X. Yin, "Recent advancements of robotics in construction," *Automation in Construction*, vol. 144, p. 104591, Dec. 2022.

[4] M. Gharbia, A. Chang-Richards, Y. Lu, R. Y. Zhong, and H. Li, "Robotic technologies for on-site building construction: A systematic review," *Journal of Building Engineering*, vol. 32, p. 101584, Nov. 2020.

[5] N. Melenbrink, J. Werfel, and A. Menges, "On-site autonomous construction robots: Towards unsupervised building," *Automation in Construction*, vol. 119, p. 103312, Nov. 2020.

[6] G. O. Tysse, A. Cibicik, and O. Egeland, "Vision-Based Control of a Knuckle Boom Crane With Online Cable Length Estimation," *IEEE/ASME Transactions on Mechatronics*, vol. 26, pp. 416–426, Feb. 2021.

[7] E. Ayoub, H. Fernando, W. Larrivée-Hardy, N. Lemieux, P. Giguère, and I. Sharf, "Log Loading Automation for Timber-Harvesting Industry," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, May 2024, pp. 17 920–17 926.

[8] M.-P. Ecker, B. Bischof, M. N. Vu, C. Fröhlich, T. Glück, and W. Kemmetmüller, "Near time-optimal hybrid motion planning for timber cranes," in *2025 IEEE International Conference on Robotics and Automation (ICRA)*, 2025, pp. 1467–1473.

[9] J. Kalmari, J. Backman, and A. Visala, "Nonlinear model predictive control of hydraulic forestry crane with automatic sway damping," *Computers and Electronics in Agriculture*, vol. 109, pp. 36–45, Nov. 2014.

[10] P. Schubert and D. Abel, "Flatness-based Model Predictive Payload Control for Offshore Cranes," in *2023 European Control Conference (ECC)*, Jun. 2023, pp. 1–8.

[11] I. Jebellat and I. Sharf, "Motion planners for path or waypoint following and end-effector sway damping with dynamic programming," *IEEE Transactions on Automation Science and Engineering*, vol. 22, pp. 8439–8452, 2025.

[12] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics*, ser. Advanced Textbooks in Control and Signal Processing. London, UK: Springer, 2009.

[13] K. M. Lynch and F. C. Park, *Modern robotics: mechanics, planning, and control*. Cambridge, UK: Cambridge University Press, 2017.

[14] M. N. Vu, C. Hartl-Nesic, and A. Kugi, "Fast swing-up trajectory optimization for a spherical pendulum on a 7-DoF collaborative robot," in *IEEE International Conference on Robotics and Automation*, Xi'an, China, 2021, pp. 10 114–10 120.

[15] D. Q. Huynh, "Metrics for 3D Rotations: Comparison and Analysis," *Journal of Mathematical Imaging and Vision*, vol. 35, no. 2, pp. 155–164, 2009.

[16] 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.

[17] R. Featherstone, *Rigid Body Dynamics Algorithms*. New York: Springer, 2008.

[18] J. Carpentier, G. Saurel, G. Buondonno, J. Mirabel, F. Lamiraux, O. Stasse, and N. Mansard, "The Pinocchio C++ library," in *2019 IEEE/SICE International Symposium on System Integration (SII)*. Paris, France: IEEE, Jan. 2019, pp. 614–619.

[19] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi: A software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, pp. 1–36, Mar. 2019.

[20] R. Verschueren, G. Frison, D. Kouzoupis, J. Frey, N. van Duijkeren, A. Zanelli, B. Novoselnik, T. Albin, R. Quirynen, and M. Diehl, "Acados—a modular open-source framework for fast embedded optimal control," *Mathematical Programming Computation*, vol. 14, pp. 147–183, Mar. 2022.

[21] M. Diehl, H. G. Bock, and J. P. Schlöder, "A real-time iteration scheme for nonlinear optimization in optimal feedback control," *SIAM Journal on control and optimization*, vol. 43, no. 5, pp. 1714–1736, 2005.

[22] G. Frison and M. Diehl, "HPIPM: A high-performance quadratic programming framework for model predictive control," *IFAC-PapersOnLine*, vol. 53, pp. 6563–6569, Jan. 2020.

[23] N. T. Nguyen, L. Schilling, M. S. Angern, H. Hamann, F. Ernst, and G. Schildbach, "B-spline path planner for safe navigation of mobile robots," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep. 2021, pp. 339–345.

[24] T. Mercy, R. Van Parys, and G. Pipeleers, "Spline-Based Motion Planning for Autonomous Guided Vehicles in a Dynamic Environment," *IEEE Transactions on Control Systems and Technology*, vol. 26, pp. 2182–2189, Nov. 2018.

[25] G. Williams, A. Aldrich, and E. A. Theodorou, "Model predictive path integral control: From theory to parallel computation," *Journal of Guidance, Control, and Dynamics*, vol. 40, no. 2, pp. 344–357, 2017.

[26] P.-T. de Boer, D. P. Kroese, S. Mannor, and R. Y. Rubinstein, "A tutorial on the cross-entropy method," *Annals of Operations Research*, vol. 134, no. 1, pp. 19–67, 2005.

[27] M. N. Vu, G. Ebmer, A. Wachter, M.-P. Ecker, G. Nguyen, and T. Glueck, "GPU-accelerated motion planning of an underactuated forestry crane in cluttered environments," *IFAC-PapersOnLine*, vol. 59, no. 18, pp. 295–300, 2025.

[28] E. Todorov, T. Erez, and Y. Tassa, "MuJoCo: A physics engine for model-based control," in *2012 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, October 2012, pp. 5026–5033.

[29] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and M. J. Marín-Jiménez, "Automatic generation and detection of highly reliable fiducial markers under occlusion," *Pattern Recognition*, vol. 47, no. 6, pp. 2280–2292, 2014.

[30] T. Collins and A. Bartoli, "Infinitesimal plane-based pose estimation," *International Journal of Computer Vision*, vol. 109, pp. 252–286, 2014.

[31] E. J. Lefferts, F. L. Markley, and M. D. Shuster, "Kalman filtering for spacecraft attitude estimation," *Journal of Guidance, Control, and Dynamics*, vol. 5, no. 5, pp. 417–429, 1982.

[32] F. L. Markley, "Attitude error representations for kalman filtering," *Journal of Guidance, Control, and Dynamics*, vol. 26, no. 2, pp. 311–317, 2003.

[33] S. O. H. Madgwick, A. J. L. Harrison, and R. Vaidyanathan, "Estimation of IMU and MARG orientation using a gradient descent algorithm," in *2011 IEEE International Conference on Rehabilitation Robotics*, June 2011, pp. 1–7.

[34] R. Ros and N. Hansen, "A simple modification in cma-es achieving linear time and space complexity," in *International conference on parallel problem solving from nature*. Springer, September 2008, pp. 296–305.