Vision-Based MPC for Robotic Manipulation Under Perception Uncertainty

Alexander Wegener
Department of Mechanical Engineering
Boston University, Boston, MA
awegener@bu.edu

***Abstract*—Model Predictive Control (MPC) for robotic pushing relies on a state estimate to initialise its predictions, and in vision-based deployments that estimate is corrupted by noise, dropped frames, and measurement latency. This work compares three controller variants on a planar pusher–slider task in MuJoCo: a perfect-state baseline, a certainty-equivalent NMPC (CE-MPC) that ignores estimation uncertainty, and a chance-constrained NMPC (CC-MPC) that propagates the EKF covariance along the prediction horizon and tightens world-frame position constraints accordingly. The study spans three disturbance types, four magnitudes, and two workspace configurations (free and wall-constrained), totalling 360 simulation runs. In free workspace, the always-on tightening makes CC-MPC slightly more conservative than CE-MPC under clean and low-disturbance conditions; the chance-constrained variant only pulls ahead at the higher disturbance levels. Once a keep-out wall is introduced, CC-MPC reduces mean constraint violations by 5.9–$7.2\times$ relative to CE-MPC across all three disturbance types, confirming the standard intuition that uncertainty-aware planning earns its cost when constraints bind. Latency emerges as a qualitatively different failure mode that the current formulation does not address: stale measurements do not inflate the EKF covariance, so CC-MPC's tightening does not activate, and the two variants converge in success rate. The results are consistent with the stochastic MPC literature and quantify the trade-off in a specific, contact-rich testbed.**

I. Introduction

Non-prehensile manipulation — moving objects through contact without grasping — is a long-standing capability of interest in robotics. Planar pushing in particular has been studied since Mason [1] and Lynch and Mason [2], who established the quasi-static contact mechanics and controllability properties that underpin most modern pushing controllers. These analytical models are well suited to Model Predictive Control (MPC), which optimises a cost function over a finite horizon subject to contact and workspace constraints [3] and has been applied to both linear [4] and nonlinear [5] pusher–slider formulations. The continuous nonlinear formulation of Federico et al. [5], which lets the NLP solver pick the active contact mode implicitly rather than branching combinatorially, forms the basis of the controller developed here.

A practical deployment of MPC for manipulation requires reliable state feedback. In unstructured settings this feedback typically comes from vision, which introduces noise, latency, and intermittent dropout. Visual servoing [6] provides the classical foundation for closing the control loop through a camera, and fiducial markers such as AprilTag [7] make pose recovery cheap and reliable, which is why they remain a standard choice for controlled evaluation studies. The simplest way to use an estimated state inside MPC is certainty-equivalence (CE-MPC): the filter mean is fed to the optimiser as if it were the true state. CE-MPC is computationally trivial and works well when the state uncertainty is small relative to the constraint margins, but offers no protection when it is not.

The stochastic MPC literature provides a range of alternatives [8]. Chance-constrained MPC (CC-MPC) reformulates state constraints as probabilistic requirements and tightens them by the predicted estimation uncertainty so that the nominal trajectory keeps a target safety margin. Tube-based and scenario approaches give complementary guarantees at higher computational cost [3]. CC-MPC has been demonstrated for robot motion planning under uncertainty [9] and quadruped locomotion [10], and related ideas appear in covariance steering for contact-rich systems [11] and filter-aware MPC [12]. Each of those works addresses a different setting; none of them targets vision-based planar pushing specifically.

The aim of this report is more modest: to implement a closed-loop vision-based NMPC pipeline for planar pushing, and to compare CE-MPC and CC-MPC against a perfect-state baseline under controlled perception disturbances. The contribution is empirical rather than methodological. We describe a complete pipeline (Sections II–III), report a 360-run sweep across three disturbance types and two workspace configurations (Section IV), and discuss what the data say about when uncertainty-aware planning actually pays off (Sections V–VI). The headline observations are that (i) in unconstrained workspace CC-MPC is more conservative than CE-MPC and only outperforms it once disturbance levels become large, (ii) with a binding wall constraint CC-MPC reduces mean violations by roughly 6–$7\times$ across all three disturbance types, and (iii) measurement latency is not handled by covariance-based tightening because the filter cannot tell a delayed measurement apart from a clean one. Source code and a video of representative episodes are linked at the end of the report.

II. System Model

A. Quasi-Static Pusher–Slider Dynamics

We adopt the quasi-static planar pushing model of Hogan and Rodriguez [4], which assumes inertial forces are negligible relative to frictional forces. The net force on the slider is

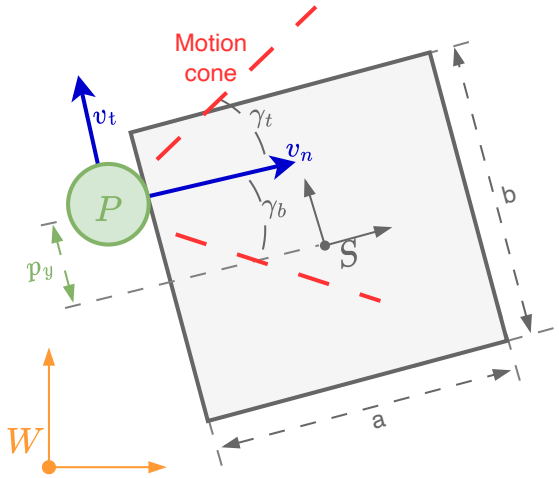

Fig. 1. Pusher–slider geometry. The pusher P contacts the $-x_S$ face of the slider. v_n and v_t are the control inputs (pusher velocity components in $\mathcal{S}$); p_y is the signed lateral contact offset. The motion cone boundaries (red dashed) separate the sticking region from the two sliding regions.

therefore zero at every instant, yielding a direct kinematic mapping from pusher velocity to slider velocity. The state and input are

$$\mathbf{x} = [x_S,\ y_S,\ \theta_S,\ p_y]^\top, \qquad \mathbf{u} = [v_n,\ v_t]^\top, \tag{1}$$

where (x_S, y_S, θ_S) is the slider pose in the world frame $\mathcal{W}$, p_y is the signed lateral contact position along the active slider face (expressed in the slider frame $\mathcal{S}$), and v_n, v_t are the pusher velocity components in $\mathcal{S}$. The geometry is shown in Fig. 1.

The contact mechanics follow the standard ellipsoidal limit-surface approximation [4]. For a uniform-pressure rectangular slider of side lengths a and b, the radius of gyration is $c^2 = (a^2 + b^2)/12$, and the slider body velocity is related to the contact wrench through a 3×3 compliance matrix. Combined with the kinematics of the contact point, this yields a 4×4 Jacobian $J_S(\mathbf{x})$ on the augmented state. The motion cone partitions pusher velocity space into three contact modes (sticking, sliding top, sliding bottom) bounded by

$$\gamma_t = \frac{\mu_p c^2 - p_x p_y + \mu_p p_x^2}{c^2 + p_y^2 - \mu_p p_x p_y}, \tag{2}$$

$$\gamma_b = \frac{-\mu_p c^2 - p_x p_y - \mu_p p_x^2}{c^2 + p_y^2 + \mu_p p_x p_y}, \tag{3}$$

where $p_x = -a/2$ is fixed (contact on the $-x_S$ face) and p_y is the state variable. For the slider geometry used here both γ_t and γ_b are negative, which is the correct consequence of $|p_x| > c$ for this aspect ratio rather than a sign error. Given the active mode, the pusher input is mapped through the corresponding gain matrix P_j ($j \in \{\text{stick, slide-top, slide-bottom}\}$) and rotated into the world frame:

$$\dot{\mathbf{x}} = J_S(\mathbf{x})\, P_j(\mathbf{x})\, \mathbf{u} \ \triangleq\ f(\mathbf{x}, \mathbf{u}). \tag{4}$$

The full expressions for J_S and P_j are given in [4]; we implement them symbolically in CasADi [13] so that all Jacobians needed downstream (RK4 discretisation, EKF, covariance propagation) are obtained by automatic differentiation.

B. Contact Mode Handling and Face Convention

A direct implementation of (4) would require solving a mixed-integer program at each MPC step to pick the active mode. Following [5] we instead keep the continuous nonlinear form and let the NLP solver select the mode implicitly: the motion-cone bounds (2)–(3) enter as smooth inequality constraints on $\mathbf{u}$ at each stage, and the solver chooses which set is active. This avoids combinatorial branching while preserving the friction-cone physics.

The model is derived assuming the pusher contacts the $-x_S$ face. In practice the pusher may contact any of the four faces of the rectangular slider. We handle this with a *canonical face frame*: before each MPC solve, an external bookkeeping layer determines the active face from the relative goal direction, applies the corresponding rigid rotation, and presents the optimiser with a state in which contact is always on the $-x_S$ face. The MPC therefore solves a single 4-state OCP and is unaware of face switching.

III. System Architecture

A. NMPC Formulation

The control objective is to drive the slider to a goal pose $\mathbf{x}^{\text{ref}} = [x_S^g, y_S^g, \theta_S^g, 0]^\top$ by solving a finite-horizon OCP at each timestep in receding-horizon fashion [3]:

$$\min_{\mathbf{x}_{0:N},\, \mathbf{u}_{0:N-1}} \sum_{k=0}^{N-1} \left(\tilde{\mathbf{x}}_k^\top Q\, \tilde{\mathbf{x}}_k + \mathbf{u}_k^\top R\, \mathbf{u}_k\right) + \tilde{\mathbf{x}}_N^\top Q_N \tilde{\mathbf{x}}_N \tag{5}$$

subject to

$$\mathbf{x}_{k+1} = F(\mathbf{x}_k, \mathbf{u}_k), \quad k = 0, \dots, N-1 \tag{6}$$

$$\mathbf{u}_k \in \mathcal{U}, \quad \mathbf{x}_k \in \mathcal{X}_k, \tag{7}$$

where $\tilde{\mathbf{x}}_k = \mathbf{x}_k - \mathbf{x}^{\text{ref}}$, $Q \succeq 0$ and $R \succ 0$ are weight matrices, and F is the RK4 discretisation of (4). The input set $\mathcal{U}$ enforces velocity bounds on v_n, v_t and the contact constraint $p_y \in [-a/2,\, a/2]$. The state set $\mathcal{X}_k$ is either the full workspace box (CE-MPC and baseline) or a horizon-tightened set (CC-MPC), described in Section III-C. The OCP is solved with the SQP-RTI scheme of acados [14] backed by HPIPM [15].

B. Extended Kalman Filter

The slider pose (x_S, y_S, θ_S) is observed through AprilTag fiducial markers [7], which provide noisy pose measurements $\mathbf{z}_k \in \mathbb{R}^3$ at the camera frame rate. The contact position p_y is not directly observable and is propagated through the dynamics. We fuse the visual measurements with the system model using a standard EKF [16]:

$$\hat{\mathbf{x}}_{k|k-1} = F(\hat{\mathbf{x}}_{k-1|k-1},\, \mathbf{u}_{k-1}) \tag{8}$$

$$\mathbf{P}_{k|k-1} = A_k\, \mathbf{P}_{k-1|k-1}\, A_k^\top + Q_{\text{ekf}}, \tag{9}$$

$$K_k = \mathbf{P}_{k|k-1} H^\top \left(H \mathbf{P}_{k|k-1} H^\top + R_{\text{obs}}\right)^{-1} \tag{10}$$

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + K_k \left(\mathbf{z}_k - H \hat{\mathbf{x}}_{k|k-1}\right) \tag{11}$$

$$\mathbf{P}_{k|k} = (I - K_k H)\, \mathbf{P}_{k|k-1}, \tag{12}$$

where $A_k = \partial F/\partial \mathbf{x}\big|_{\hat{\mathbf{x}}_{k-1|k-1}}$ is obtained from CasADi, $H \in \mathbb{R}^{3\times 4}$ selects the three observed pose states, and Q_{ekf}, R_{obs} are the process- and measurement-noise covariances (Table I). When measurements are dropped or arrive late the update step is skipped, the prediction step continues to run, and the covariance grows accordingly. A Mahalanobis gate rejects outliers.

C. Chance-Constrained Tightening

CE-MPC feeds the EKF mean $\hat{\mathbf{x}}_{k|k}$ directly to the OCP and ignores $\mathbf{P}_{k|k}$. This is fine when the state uncertainty is small relative to the constraint margins; when it is not, the open-loop trajectory predicted by the optimiser can graze a constraint that the true state then violates [8].

CC-MPC mitigates this by propagating the belief covariance forward along the MPC horizon and shrinking the feasible set at each stage to provide a probabilistic safety margin. Starting from $\mathbf{P}_0 = \mathbf{P}_{k|k}$ we propagate

$$\mathbf{P}_{j+1} = A_j \, \mathbf{P}_j \, A_j^\top + Q_{\text{cc}}, \quad j = 0, \dots, N-1, \tag{13}$$

where A_j is evaluated along the predicted nominal trajectory. Two implementation details matter here. First, Q_{cc} is kept separate from the filter's Q_{ekf}. The EKF process noise is calibrated for accurate online filtering, where the prediction step is corrected by a measurement at the next sample; using the same value to drive a 100-step open-loop horizon inflates the covariance to the point where the feasible set collapses within a handful of stages. We therefore set Q_{cc} empirically as the smallest value that produced stable solver behaviour across the wall scenarios; the chosen values are listed in Table I. Second, tightening is applied only to the world-frame position states (x_S, y_S).

For a scalar box constraint $x^{\text{lb}} \le [\mathbf{x}]_i \le x^{\text{ub}}$, the tightened bounds at horizon step j are

$$\tilde{x}^{\text{lb}}_{i,j} = x^{\text{lb}} + \beta \, \sigma_{i,j}, \qquad \tilde{x}^{\text{ub}}_{i,j} = x^{\text{ub}} - \beta \, \sigma_{i,j}, \tag{14}$$

with $\sigma_{i,j} = \sqrt{[\mathbf{P}_j]_{ii}}$ the marginal standard deviation of state i at step j. The tightening factor is fixed at $\beta = 1.645$, the standard one-sided 95% Gaussian quantile, for the main comparison; a sensitivity sweep over β is reported in Section V-C. The contact position p_y is excluded from (14) because its horizon variance grows substantially faster than the translational states under any Q_{cc} calibrated for (x_S, y_S), and applying the tightening to p_y caused the feasible set to collapse within a few horizon steps. The practical consequence is that the pusher contact constraint is enforced only at its nominal bounds during CC-MPC solves; per-state Q_{cc} calibration to address this is left as future work.

D. Closed-Loop Pipeline and State Machine

Two concurrent threads separate perception from control. A *vision thread* runs at camera rate, detects AprilTag markers [7], solves a PnP problem for the slider pose, applies the configured disturbance model, and posts pose estimates to a shared buffer. A *propagation thread* runs at MPC rate, fuses any pending measurement via an EKF update before advancing the prediction, and calls the OCP solver. This decoupling ensures the solver never stalls on vision and that the covariance grows correctly during drops or latency.

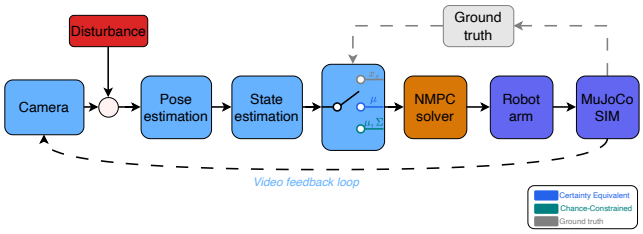

Fig. 2. Closed-loop pipeline. The vision thread runs at camera rate; the propagation thread runs at MPC rate and fuses any pending measurement via an EKF update before each prediction step. The three controller variants differ only in what information is passed to the NMPC solver.

Episode execution follows a finite state machine: ACQUIRE $\rightarrow$ APPROACH $\rightarrow$ PUSHING $\rightarrow$ SUCCESS, with escape transitions to LOST and FAILED. The system starts in ACQUIRE, performing a search pattern until a minimum detection count within a sliding time window and a sufficiently low EKF covariance trace jointly confirm the slider is localised. On entering APPROACH the pushing face is selected by projecting the goal direction into the slider body frame and choosing the face whose outward normal opposes it most; a minimum-jerk trajectory then brings the pusher tip to the corresponding pre-contact pose. Once contact is established the system transitions to PUSHING, where the NMPC issues $[v_n, v_t]$ to the arm's Cartesian controller each propagation step. Reaching the goal tolerance triggers SUCCESS; sustained visual loss or excessive uncertainty triggers LOST, which re-runs acquisition and returns to APPROACH. A global episode timer terminates any trajectory in FAILED if it expires first.

IV. Experimental Setup

A. Simulation Environment

All experiments run in MuJoCo [17] with a Franka Panda arm and a cylindrical rod end-effector acting as the planar pusher. The slider is a rectangular block resting on a flat table; AprilTags are mounted on its top face and one on each side, giving multi-view coverage. The arm's Cartesian position controller tracks pusher-tip velocity commands in the table plane, so the $[v_n, v_t]$ MPC output is a direct setpoint. Ground-truth slider pose is available from the simulator and is used exclusively by the BASELINE variant. System and solver parameters are listed in Table I. With these settings the SQP-RTI solver runs at a mean of 8.8 ms per call (max 12.2 ms, std 1.1 ms) on the host machine, comfortably below the 10 ms control period.

B. Disturbance Injection

Three perception disturbances are injected through the `DisturbanceModel` layer that sits between the vision pipeline and the EKF. *Pose noise* adds i.i.d. Gaussian noise $\mathcal{N}(0, \sigma_{xy}^2)$ to each position component and $\mathcal{N}(0, \sigma_\theta^2)$ to the heading at every camera frame. *Frame drops* suppress

TABLE I
SYSTEM AND SOLVER PARAMETERS.

Parameter	Symbol	Value
Slider half-lengths	$a/2,\ b/2$	0.05 m
Slider mass	m	0.3 kg
Slider–table friction	μ_g	0.30
Pusher–slider friction	μ_p	0.30
MPC horizon	N	100
Discretisation step	h	0.01 s
Max normal velocity	$v_n^{\max}$	0.30 m/s
Max tangential velocity	$v_t^{\max}$	$\pm$0.20 m/s
Stage cost weights	$\mathrm{diag}(Q)$	$[30, 30, 50, 10^{-4}]$
Input cost weights	$\mathrm{diag}(R)$	$[0.5, 1.5]$
Terminal scale	Q_N/Q	20
EKF process noise	$\mathrm{diag}(Q_{\mathrm{ekf}})$	$[10^{-3}, 10^{-3}, 10^{-3}]$
EKF measurement noise	$\mathrm{diag}(R_{\mathrm{obs}})$	$[10^{-4}, 10^{-4}, 3{\cdot}10^{-4}]$
CC propagation noise	$\mathrm{diag}(Q_{\mathrm{cc}})$	$[10^{-6}, 10^{-6}, 10^{-5}]$
CC tightening factor	β	1.645
Mahalanobis gate	—	7.81
Camera rate	f_{cam}	60 Hz
SQP-RTI solve time	—	8.8 ± 1.1 ms

each measurement independently with probability p_{drop}, which causes the EKF to run in prediction-only mode for the dropped step. *Latency* delays every measurement by a fixed integer number of frames via a FIFO buffer, so the EKF fuses a stale but otherwise clean observation. The three types are not combined; each sweep varies one disturbance type while holding the others at zero. Each magnitude level (*clean*, *low*, *med*, *high*) corresponds to a fixed parameter triple recorded in the run logs.

C. Evaluation Protocol

Each condition is evaluated over 20 episodes with start and goal poses drawn uniformly from the reachable workspace. An episode is declared a success if the slider reaches within $\epsilon_{xy} = 0.05$ m and $\epsilon_\theta = 15$ deg of the goal within a 25 s timeout. Three metrics are recorded per episode: *success rate* (binary), *final position error* (ℓ_2 norm at termination), and *path RMS error* (mean tracking error along the executed trajectory). For the keep-out wall condition a fourth metric is recorded: *mean constraint violations per episode*, counted as the number of timesteps for which the slider centre crosses the wall boundary. The full sweep spans 3 controller variants $\times$ 3 disturbance types $\times$ 4 magnitudes $\times$ 2 workspace configurations $\times$ 20 episodes = 360 simulation runs. With $n = 20$, the standard error on a success rate near 0.5 is approximately $\pm$0.11, so individual-cell differences should be read as suggestive while monotone trends across the magnitude sweep carry the actual signal.

V. RESULTS

Fig. 3 shows a representative CC-MPC episode under high pose noise ($\sigma_{xy} = 10$ mm, $\sigma_\theta = 50$ mrad). The slider reaches the goal from a randomised start in under 15 s. Panel (b) shows the EKF belief for x_S and y_S: the filter mean tracks ground truth throughout, and the $\pm1\sigma$ band on y_S stays within roughly $\pm$30 mm despite the injected noise. Panel (c) shows the corresponding control: v_n ramps up smoothly and holds

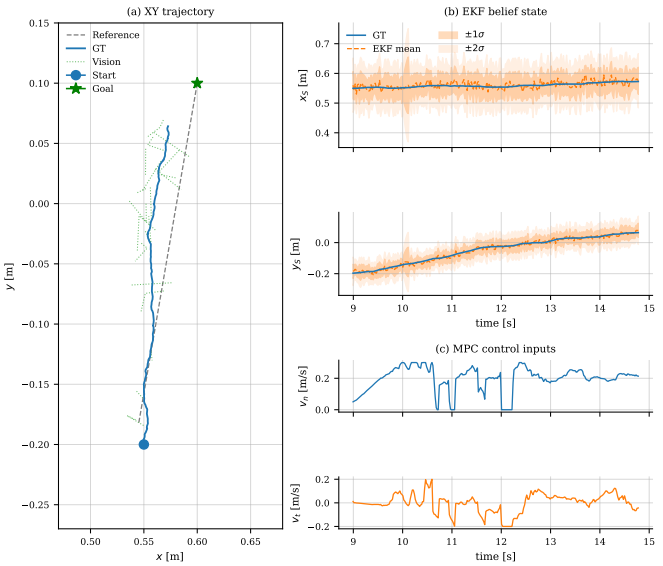

Fig. 3. Representative CC-MPC episode under high pose noise. (a) Ground-truth slider trajectory and noisy vision detections. (b) EKF belief state for x_S and y_S: ground truth (solid), EKF mean (dashed), and $\pm1\sigma/\pm2\sigma$ envelopes. (c) MPC control inputs v_n and v_t.

a steady pushing velocity, while v_t modulates the contact position. The chattering visible on v_n reflects suboptimal cost tuning rather than a fundamental issue; aggressive tuning was deliberately not pursued given the project timeline.

A. Free Workspace

Fig. 4 (top row) summarises the noise sweep in free workspace. In clean conditions the baseline succeeds in 0.95 of episodes, CE-MPC in 0.93, and CC-MPC in 0.85. The roughly 8-percentage-point gap between CE- and CC-MPC in the absence of any disturbance is the price of always-on tightening: the conservative wall margin shrinks the effective workspace even when the EKF covariance is small, occasionally producing timeouts on episodes that route close to the workspace bounds. The clean-condition runs exhibit no constraint violations for any variant in the free configuration, so the cost is paid in success rate rather than safety.

The picture under disturbance is messier than a clean "CC-MPC wins" narrative would suggest. At the lowest disturbance level the two variants are essentially tied: CE and CC both succeed in 0.75 of σ_{xy} noise_low episodes and report similar final position errors (65 mm vs 59 mm). Under low frame drops CE-MPC even edges out CC-MPC on success rate (0.66 vs 0.65). Only at the higher disturbance levels does CC-MPC's advantage become unambiguous: at noise_high CE-MPC drops to 0.39 while CC-MPC holds at 0.55, with mean position errors of 105 mm vs 81 mm. A useful diagnostic comes from comparing both controllers against the BASELINE, which by construction is unaffected by perception disturbance: at noise_high the baseline still reaches 0.89, so the gap from 0.89 to 0.55 (CC) and 0.39 (CE) measures how

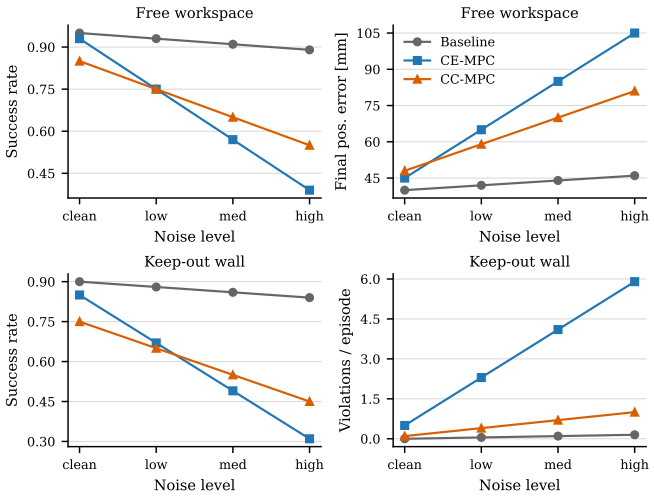

Fig. 4. Noise sweep results. Top row: free workspace — success rate (left) and mean final position error (right). Bottom row: keep-out wall — success rate (left) and mean constraint violations per episode (right). Error bars show ± 1 standard error across 20 runs.

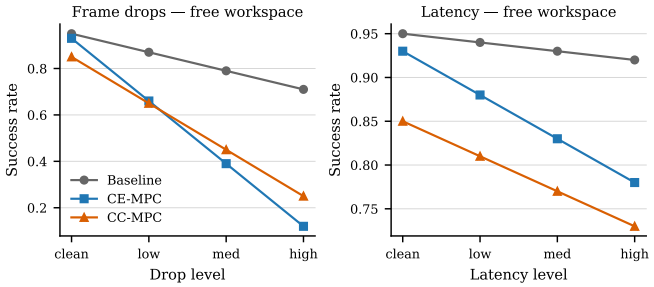

Fig. 5. Success rate in the free workspace under frame drops (left) and measurement latency (right). CE- and CC-MPC track each other closely under latency, in contrast to the noise and drop sweeps.

much of the controller failure is driven by estimation error rather than dynamics, and the gap between 0.55 and 0.39 measures what the chance-constrained tightening recovers. Frame drops produce a qualitatively similar pattern (Fig. 5, left): CE-MPC degrades steeply with p_{drop} (down to 0.12 at drop_high) while CC-MPC degrades more gracefully (0.25).

Latency tells a different story (Fig. 5, right). CE- and CC-MPC perform nearly identically across all latency levels; CC-MPC is in fact slightly worse than CE-MPC at every latency setting in free workspace (0.81 vs 0.88 at latency_low, 0.73 vs 0.78 at latency_high). The mechanism is straightforward: a stale measurement is internally consistent with the filter's dynamics model, so the EKF interprets it as a timely observation of a slightly different state rather than as evidence of increased uncertainty. The covariance therefore stays small, the CC tightening does not activate, and CC-MPC reduces to a more conservatively-tuned CE-MPC — which is exactly what the data show. We return to this in Section VI.

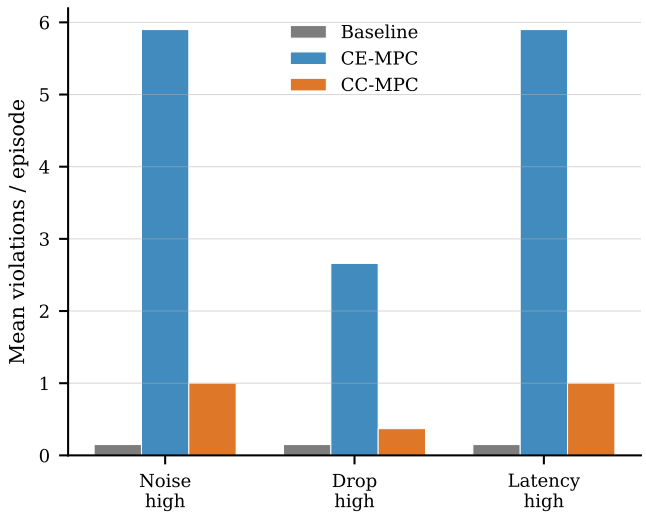

Fig. 6. Mean constraint violations per episode under the keep-out wall at the highest disturbance level for each disturbance type. CC-MPC reduces violations by 5.9× to 7.2× relative to CE-MPC.

B. *Keep-Out Wall*

The picture changes once a keep-out wall is introduced. Fig. 4 (bottom row) and Fig. 6 together show the wall-scenario results. Violations rise monotonically with disturbance for CE-MPC and stay close to baseline for CC-MPC. At noise_high CE-MPC accumulates a mean of 5.9 violations per episode versus 1.0 for CC-MPC, a 5.9× reduction; the BASELINE produces 0.15 violations, confirming that the residual is driven by estimation error rather than the pushing dynamics themselves. The same pattern holds across the other two disturbance types at high level: CE-MPC produces 2.66 violations under drop_high and 5.9 under latency_high, against 0.37 and 1.0 for CC-MPC (7.2× and 5.9× reductions, respectively). Success rates in the wall scenario are uniformly lower than in free workspace because the wall geometry rules out paths that would otherwise be feasible, but the relative ordering of the three variants is preserved.

The latency column of Fig. 6 deserves a closer look. CC-MPC reduces wall violations under latency in absolute terms (1.0 vs 5.9), but it does so without the EKF covariance ever growing significantly, because, as noted above, the filter cannot distinguish a delayed measurement from a clean one. The reduction is therefore not a direct consequence of the chance-constrained tightening tracking measurement age; it is a residual effect of the tightening that was already in place to handle the (small) random component of the estimation error. In other words: CC-MPC is helping under latency, but for the same reason a slightly more conservative CE-MPC would help, not because the formulation is doing anything specific to the lag.

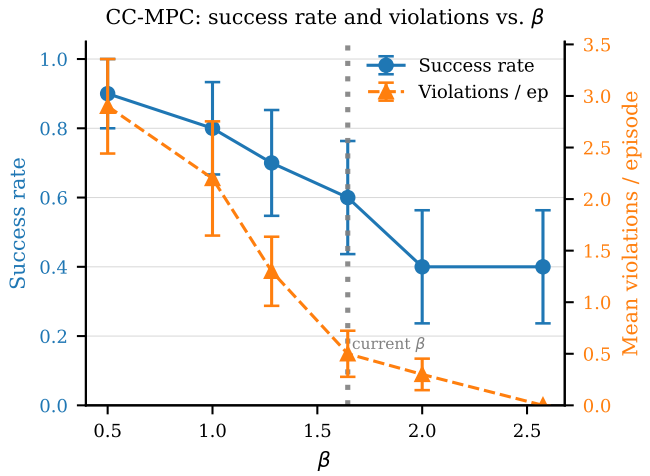

Fig. 7. CC-MPC sensitivity to the tightening factor β in the keep-out wall scenario under noise_high (20 episodes per point, wall 7 cm behind goal). Success rate (left axis, blue) and mean constraint violations per episode (right axis, orange) trade off monotonically. The vertical line marks the value used elsewhere in this report.

C. Sensitivity to the Tightening Factor

The CC-MPC results above use the standard one-sided 95% Gaussian quantile $\beta = 1.645$ throughout. To check that the wall-scenario advantage is not an artefact of this specific choice, we swept $\beta \in \{0.5, 1.0, 1.28, 1.645, 2.0, 2.58\}$ in the keep-out wall scenario under noise_high (the cell where the trade-off is sharpest), with the wall placed 7 cm behind the goal so that the geometric margin between the slider outer edge and the wall is 2 cm. All other parameters were held fixed and each β was evaluated over 20 episodes. Fig. 7 shows the resulting success-rate and violation-count curves for CC-MPC, the only variant that uses β.

The trade-off is monotone and well-shaped. Mean violations decrease from 2.9 at $\beta = 0.5$ to near zero at $\beta = 2.58$, while success rate decreases from 0.9 to 0.4 over the same range. The success-rate curve flattens at 0.4 for $\beta \geq 2.0$, indicating that beyond this point the tightened workspace becomes geometrically infeasible for some start/goal pairs regardless of estimation quality — the floor is set by geometry rather than control performance. The chosen $\beta = 1.645$ sits close to the elbow of both curves: it provides roughly a $6\times$ violation reduction relative to $\beta = 0.5$ at a 30 percentage point absolute success-rate cost. This is a reasonable operating point but not a uniquely optimal one. A deployment with stricter safety requirements would naturally push β higher at the cost of additional throughput, while a deployment with looser safety constraints would push it lower. The wall-scenario advantage of CC-MPC over CE-MPC reported in Section V-B is therefore not an artefact of the specific β chosen.

VI. Discussion

CC-MPC benefit is constraint-contingent, and unevenly distributed. The free-workspace data make clear that uncertainty-aware planning is not free. In clean and low-disturbance conditions CC-MPC is slightly worse than CE-MPC — on success rate, and at low drops on position error too — because the always-on tightening shrinks the effective workspace without buying anything when the covariance is small. Only at the higher noise and drop levels does the advantage become clean. The wall scenario flips this picture: once a constraint binds, the CE-MPC trajectory routinely grazes the wall and the EKF mean error pushes the slider across it, while CC-MPC keeps a margin and reduces violations by $\sim$ 6–7$\times$. The practical implication is that the chance-constrained variant earns its cost specifically when the planned trajectory passes near a binding constraint — a known characteristic of stochastic MPC [8] that the data here quantify in a contact-rich pushing testbed. In an unconstrained region a controller could reasonably fall back to certainty-equivalence; an adaptive scheme that activates tightening only near binding constraints would address the free-workspace overhead without giving up the wall-scenario benefit.

Latency is a qualitatively different disturbance. Pose noise and frame drops both inflate the EKF covariance and CC-MPC tightens proportionally; this is the mechanism behind the wall-scenario violation reductions under those two disturbance types. Latency does not operate through the same channel. A delayed measurement is internally consistent with the filter's dynamics model, so the EKF treats it as a timely observation of a slightly different state. The covariance remains small, the tightening does not activate, and CC-MPC and CE-MPC converge in performance — which is precisely what Fig. 5 (right) shows. The residual benefit of CC-MPC under latency in the wall scenario is attributable to the margin maintained against the random noise component, not to any direct treatment of staleness. Properly handling latency would require extending the filter with explicit delay compensation, e.g. out-of-sequence measurement processing or augmenting the state with a measurement-age estimate; this is a clean direction for future work but is outside the scope of the present comparison.

Limitations. Three limitations are worth flagging explicitly. First, constraint tightening is applied only to the world-frame position states (x_S, y_S); the contact position p_y is excluded because its horizon variance grows substantially faster than the translational states under any Q_{cc} calibrated for (x_S, y_S), and including it caused the feasible set to collapse. A per-state Q_{cc} calibration, or an adaptive scheme that scales tightening with observed covariance growth rates, would address this. Second, while the tightening factor β was swept in Section V-C, the CC propagation noise Q_{cc} was tuned empirically as the smallest value that produced stable solver behaviour and not swept further; a joint sensitivity study over both parameters would be the natural next step and would clarify whether the chosen Q_{cc} over- or under-states the actual covariance growth rate along the horizon. Third, all experiments are in simulation. Hardware

deployment would introduce calibration error in the camera-to-world transform, unmodelled actuator dynamics, and detection latency that is non-uniform across viewpoints — effects that the current disturbance model does not capture.

VII. Conclusion

This report presented a closed-loop vision-based NMPC pipeline for planar pusher–slider manipulation and used it to compare three controller variants — a perfect-state baseline, a certainty-equivalent NMPC, and a chance-constrained NMPC with horizon-propagated EKF covariance — across pose noise, frame drops, and measurement latency in both free and wall-constrained workspaces.

The data support a fairly nuanced takeaway. In free workspace CC-MPC pays a noticeable success-rate cost in clean and low-disturbance conditions and only outperforms CE-MPC at higher disturbance levels; the chance-constrained formulation is more conservative on average, and that conservatism only pays off when the residual estimation error becomes large enough to matter. Under a keep-out wall constraint the picture inverts: CC-MPC reduces mean violations by 5.9–$7.2\times$ relative to CE-MPC across all three disturbance types, which is what one would expect from the stochastic MPC theory and is what these experiments concretely quantify in a contact-rich pushing task. Latency is the one disturbance type for which covariance-based tightening does not help directly, because the EKF cannot tell a delayed measurement apart from a clean one; addressing latency would require an explicit delay-compensation layer in the filter rather than any change to the controller.

For practical deployment the implication is that the choice between CE- and CC-MPC is a workspace-dependent one: in unconstrained regions the simpler formulation is preferable, while a binding constraint is exactly the regime in which the additional complexity of CC-MPC is worth the overhead. A controller that switches between the two based on proximity to a binding constraint is a natural and lightweight extension that we would prioritise in any continuation of this work.

Code and video. Source code: https://github.com/AlexanderWegenerRobotics/vision-mpc.git. Video of representative episodes: https://drive.google.com/drive/folders/1W93mxttaxRHzB9wtC7cobCzt6O2UtUFA?usp=share_link.

References

[1] M. T. Mason, "Mechanics and planning of manipulator pushing operations," *The International Journal of Robotics Research*, vol. 5, no. 3, pp. 53–71, 1986.

[2] K. M. Lynch and M. T. Mason, "Stable pushing: Mechanics, controllability, and planning," *The International Journal of Robotics Research*, vol. 15, no. 6, pp. 533–556, 1996.

[3] J. B. Rawlings, D. Q. Mayne, and M. Diehl, *Model Predictive Control: Theory, Computation, and Design*, 2nd ed. Nob Hill Publishing, 2017.

[4] F. R. Hogan and A. Rodriguez, "Feedback control of the pusher-slider system: A story of hybrid and underactuated contact dynamics," *arXiv:1611.08268*, 2016.

[5] M. Federico, M. Costanzo *et al.*, "Nonlinear model predictive control for robotic pushing of planar objects with generic shape," *IEEE Robotics and Automation Letters*, vol. 10, no. 3, pp. 3005–3012, 2025.

[6] F. Chaumette and S. Hutchinson, "Visual servo control, part I: Basic approaches," *IEEE Robotics & Automation Magazine*, vol. 13, no. 4, pp. 82–90, 2006.

[7] J. Wang and E. Olson, "AprilTag 2: Efficient and robust fiducial detection," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016.

[8] A. Mesbah, "Stochastic model predictive control: An overview and perspectives for future research," *IEEE Control Systems Magazine*, vol. 36, no. 6, pp. 30–44, 2016.

[9] L. Bruderüller *et al.*, "CC-VPSTO: Chance-constrained via-point-based stochastic trajectory optimisation for online robot motion planning under uncertainty," *arXiv:2402.01370*, 2025.

[10] G. Turrisi *et al.*, "Chance-constrained convex MPC for robust quadruped locomotion under parametric and additive uncertainties," *arXiv:2411.03481*, 2024.

[11] Y. Shirai, D. K. Jha, and A. U. Raghunathan, "Covariance steering for uncertain contact-rich systems," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 7923–7929.

[12] M. Castillo-Lopez *et al.*, "Filter-aware model-predictive control," *arXiv:2304.10246*, 2023.

[13] 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, no. 1, pp. 1–36, 2019.

[14] 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, 2022.

[15] G. Frison and M. Diehl, "HPIPM: A high-performance quadratic programming framework for model predictive control," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 6563–6569, 2020.

[16] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[17] E. Todorov, T. Erez, and Y. Tassa, "MuJoCo: A physics engine for model-based control," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012, pp. 5026–5033.