State-Constrained Offline RL Without Actions: Certified Latent Reachability from Observation-Only Trajectories

Liz Lemma Future Detective

January 20, 2026

Abstract

State-constrained offline reinforcement learning (SC-offline RL) relaxes classic batch constraints by allowing out-of-distribution actions that lead back to in-distribution states, enabling stronger trajectory stitching. Existing algorithms (e.g., StaCQ) learn reachability using forward/inverse dynamics trained on (s,a,s') data, thus requiring action logs. In many 2026-era settings—video, passive sensors, proprietary controllers—actions are missing or untrustworthy. We formalize action-free SC-offline RL: the learner observes only observation sequences and rewards. Our core contribution is a latent controllability framework in which reachability is defined and certified in a learned latent state space via an implicit control variable. We propose a retrieve-then-certify algorithm that (i) learns a Markov latent representation, (ii) fits a latent-control forward model and an implicit inverse controller from observation pairs, (iii) constructs confidence-certified reachability sets, and (iv) performs state-constrained QSS learning over reachable dataset states. The main theoretical results (a) reduce action-free SC-offline RL to standard SCQL under an explicit identifiability condition, (b) provide an end-to-end performance bound degrading gracefully with reachability false positives and model/representation error, and (c) give a matching impossibility result showing identifiability is necessary. We outline experiments on pixel-based control benchmarks and robotic video datasets with actions withheld to validate stitching and safety.

Table of Contents

guarantees.

3. 3. Identifiability and Controllability Assumptions: formal conditions under which actions can be replaced by implicit latent controls; examples and counterexamples.

4. 4. Certified Latent Reachability: definition of reachability confidence sets, calibration targets (false-positive control), and retrieve-then-certify pipeline.

5. 5. Algorithm: Action-Free State-Constrained QSS Learning (AF-SCQ): representation learning, latent forward/inverse training, reachability graph construction, QSS training, policy deployment via inverse controller.

6. 6. Theory I (Upper Bounds): contraction and policy evaluation for the certified state-constrained Bellman operator; performance loss bound in terms of reachability FP/FN, representation error, and model error.

7. 7. Theory II (Lower Bounds / Impossibility): show that without identifiability, observation-only data cannot distinguish MDPs with different optimal policies; derive worst-case lower bound.

8. 8. Computational Complexity: scalability via approximate retrieval (ANN), certification cost, and memory/time tradeoffs; comparison to quadratic reachability enumeration.

9. 9. Experiments (Strengthening Evidence): withheld-actions evaluation on D4RL, pixel variants, video datasets; ablations on calibration, retrieval size, and representation choice; failure cases.

10. 10. Related Work and Discussion: offline RL, control from video, latent action models, bisimulation/representation metrics; implications for robotics and autonomy.

11. 11. Conclusion: summary, limitations, and next steps (multi-step reachability, partial observability, multi-agent).

1 Introduction

We consider control problems in which the available data are *passive* logs of experience: sequences of observations together with realized rewards, but without the actions that generated the transitions. This regime is not contrived. In robotics, one often has video demonstrations or teleoperation footage in which actuator commands are missing, corrupted, or not synchronized with perception. In autonomous driving, large-scale collections of sensor streams may omit low-level control due to proprietary interfaces. In medical and economic settings, the situation is more severe: the notion of an "action" may be latent (a treatment mixture, an institutional decision, an unrecorded intervention), while outcomes and covariates are logged with high fidelity. In all such cases, we have trajectories, but not the causal control channel.

Offline reinforcement learning, in its standard formulation, presupposes an offline dataset of the form (s_t, a_t, r_t, s_{t+1}). The action labels play two logically distinct roles. First, they identify which transitions are feasible under which controls, hence they determine the transition kernel of the induced decision process. Second, they permit *batch constraints*: algorithms can restrict the learned policy to remain close to the behavior policy in action space, thereby reducing extrapolation error when function approximation is used. Both roles become ambiguous when actions are unobserved. The first role is definitional: if we do not know which action produced a transition, then we do not know which choices were available. The second role is algorithmic: without an empirical action distribution, one cannot directly implement common pessimism or conservatism mechanisms that penalize out-of-distribution actions.

It is therefore natural to ask whether one can avoid explicit action modeling altogether. A classical observation is that many control objectives can be expressed in terms of *next-state choice*. If the agent can reliably decide to move from a current state to one of a set of attainable successor states, then planning can proceed by dynamic programming over these attainable successors. Actions are then merely a means of realizing a chosen successor. This suggests that, even when logged actions are missing, one might attempt to learn (i) a representation in which the process is Markovian, (ii) a notion of *reachability* between represented states, and (iii) a mechanism to realize a chosen reachable transition at deployment time.

However, this viewpoint immediately exposes a difficulty that is sometimes hidden in action-centric offline RL. If we are given only an observation stream, then the induced dataset contains transitions that occurred under some unknown behavior process. From these transitions alone, we can infer that the next observation *occurred*, but we cannot infer that it is *controllable* from the current observation. Put differently, passive logs conflate what is *possible* under some control with what is *available* to the learner as a

decision. This is an identifiability issue: two environments may generate identical observation-only datasets while differing in which transitions can be intentionally induced. Any method that plans over unverified transitions risks selecting an infeasible "shortcut" that exists only as an artifact of the logging process.

This motivates a constraint discipline that is stricter than the usual offline requirement of staying within the support of the dataset. In standard batch-constrained methods, one typically enforces that the learned policy's actions have high likelihood under the behavior action distribution, thereby hoping that the resulting state visitation remains close to the data distribution. In the action-free setting, we cannot state such a constraint. Instead, we are led to a *state-constrained* principle: the learned policy should, with high probability, select transitions whose endpoints remain within the set of states represented in the dataset (or a certified neighborhood thereof), and whose one-step feasibility is supported by a test with controlled false-positive rate. This shifts the focus from action support to *transition support* among dataset states.

State-constrained offline RL has a further conceptual advantage: it isolates the part of the problem that is genuinely learnable from passive data. If we restrict attention to planning over states that actually appear in the dataset, then we avoid value extrapolation into regions with no empirical grounding. Moreover, if we require that Bellman backups only maximize over a verified set of candidate successor states, then the chief failure mode becomes explicit: the only way to obtain spurious optimism is to admit an unreachable successor into the maximization set. Thus, the statistical object of interest is not an action-value function $Q(s, a)$ but a state-successor value $Q(s, s')$ (or its latent analogue), together with a reachability relation $s' \in \mathrm{Reach}(s)$ that formalizes which successors are executable.

The central bottleneck remains the same: we must determine whether a proposed edge (s, s') is executable by the agent. When actions are observed, one can appeal to supervised learning of dynamics and inverse dynamics, together with policy evaluation under known actions. When actions are not observed, we must infer a control variable indirectly, for example by positing a latent control space and learning an "implicit inverse model" that maps pairs (s, s') to a latent control intended to realize s'. Even if such a model is trained to match one-step transitions in the dataset, it can still hallucinate feasibility for pairs of states never connected by any controllable action. Consequently, some form of *reachability certification* is not optional if one seeks guarantees: one must bound the probability of accepting an infeasible edge.

We therefore take the following position. The proper analogue of batch-constrained offline RL in the action-free regime is a procedure that (a) learns a representation and a latent forward/inverse control model from observation transitions, (b) constructs a sparse graph over dataset states by proposing

candidate successors and *certifying* the one-step realizability of each proposed edge with calibrated confidence, and (c) performs dynamic programming over this certified graph to obtain a policy that selects among certified successors. The resulting policy is intrinsically state-constrained: it never plans to leave the dataset support in one step, except through a controlled certification slack. Its performance degrades gracefully as a function of (i) model and representation errors and (ii) the false-positive and false-negative rates of the certification step.

Finally, we emphasize an unavoidable limitation. Without an assumption ensuring that the latent control is identifiable from state transitions—or, more generally, without side information that distinguishes controllable from incidental transitions—no algorithm can guarantee vanishing suboptimality uniformly over environments consistent with the same observation-only dataset. This is not a defect of any particular method, but a consequence of indistinguishability. The appropriate goal is therefore conditional: under an identifiability and calibration regime, we can recover a near-optimal *state-constrained* policy; without such conditions, one should not expect nontrivial guarantees. The next section formalizes the observation-only setting, the induced state-constrained optimality criterion, and the target bounds we seek.

2 Problem Setup

We work in an offline setting in which the learner observes transitions and rewards but not the actions that produced them. Concretely, we are given a dataset

$$\mathcal{D} \;=\; \{(o_t, o_{t+1}, r_t)\}_{t=1}^{N},$$

where each $o_t \in \mathcal{O}$ is an observation (e.g., an image or a sensor vector) and $r_t \in \mathbb{R}$ is a realized reward. We assume the data were generated by some unknown behavior process in an underlying discounted MDP $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, r, \gamma)$, with discount factor $\gamma \in [0, 1)$. The learner may perform arbitrary computation on $\mathcal{D}$ (including representation learning and model fitting), but has no access to the action labels and no interaction with $\mathcal{M}$ during training. Optionally, a simulator may exist for evaluation only; it plays no role in defining the learning problem.

The absence of actions forces us to formulate the decision problem in a way that does not require conditioning on $a \in \mathcal{A}$. We therefore adopt a latent-state viewpoint. We posit a representation map $\phi : \mathcal{O} \to \mathcal{Z}$ and write $z_t = \phi(o_t)$. We denote by

$$\mathcal{Z}_{\mathcal{D}} \;:=\; \{z_t : (o_t, o_{t+1}, r_t) \in \mathcal{D}\}$$

the set (or empirical distribution) of latent states appearing in the dataset.

Our aim is to plan and act using only objects indexed by $z \in \mathcal{Z}_{\mathcal{D}}$, thereby avoiding extrapolation to unobserved regions of latent space.

State-constrained control without actions. If action labels were available, one would define a policy $\pi(a \mid z)$ and evaluate it via the usual return. In the action-free regime we instead treat a policy as a selector of successor latent states. Formally, a *state-constrained next-state policy* is a mapping

$$\pi_{\mathrm{SC}} : \mathcal{Z}_{\mathcal{D}} \to \Delta(\mathcal{Z}_{\mathcal{D}}),$$

where $\Delta(\mathcal{Z}_{\mathcal{D}})$ denotes distributions over dataset latent states. The semantics of $\pi_{\mathrm{SC}}(z)$ is: from current latent state z, the policy proposes a desired next state $z' \in \mathcal{Z}_{\mathcal{D}}$. Such a proposal is meaningful only if z' is *reachable* from z by some executable control in the underlying system. We therefore introduce an (unknown) *reachability relation*

$$\mathrm{Reach}(z) \subseteq \mathcal{Z}_{\mathcal{D}},$$

interpreted as the set of successor dataset latent states that can be realized from z in one step with nonnegligible probability under some available control mechanism. This object is intentionally agnostic to the original action space $\mathcal{A}$; it captures only the feasible one-step transitions among dataset states.

Given Reach, we obtain an induced decision problem on $\mathcal{Z}_{\mathcal{D}}$ in which the admissible choices at z are elements of $\mathrm{Reach}(z)$. A policy is admissible if it selects only reachable successors, i.e.,

$$\pi_{\mathrm{SC}}(z' \mid z) = 0 \quad \text{whenever} \quad z' \notin \mathrm{Reach}(z).$$

We define the optimal value among such policies by

$$V^{\pi^\star_{\mathrm{SC}}}(z) \;:=\; \sup_{\pi_{\mathrm{SC}}:\, \mathrm{supp}(\pi_{\mathrm{SC}}(\cdot|z)) \subseteq \mathrm{Reach}(z)\ \forall z} \mathbb{E}\Big[\sum_{t\geq 0} \gamma^t r_t \;\Big|\; z_0 = z\Big],$$

where the expectation is taken over the (environment-induced) evolution of latent states when the policy repeatedly proposes reachable successors. This criterion is the natural replacement for unconstrained optimal control when we commit to staying within the dataset support in one step.

QSS value functions. To implement dynamic programming over successor choices, we work with state–successor values. For any admissible policy π_{SC}, define

$$Q^{\pi_{\mathrm{SC}}}(z, z') \;:=\; \mathbb{E}\Big[r(z, z') + \gamma V^{\pi_{\mathrm{SC}}}(z_1) \;\Big|\; z_0 = z,\ z_1 = z'\Big],$$

where $r(z, z')$ denotes the reward associated with transitioning from z to z' (in practice, this may be taken as the dataset reward on observed pairs

or a learned model $\hat{r}(z, z')$). The corresponding optimal *QSS* function over reachable successors is

$$Q^{\star}_{\mathrm{SC}}(z, z') \;=\; r(z, z') + \gamma \max_{\bar{z} \in \mathrm{Reach}(z')} Q^{\star}_{\mathrm{SC}}(z', \bar{z}), \qquad z' \in \mathrm{Reach}(z),$$

and an optimal state-constrained policy may be taken as a greedy selector of z' maximizing $Q^{\star}_{\mathrm{SC}}(z, z')$ over $\mathrm{Reach}(z)$.

Learning goal. Since neither ϕ nor Reach is known, we seek a fully offline procedure that outputs a deployable controller. Operationally, deployment requires mapping the current observation o to a latent $z = \hat{\phi}(o)$ and then realizing the chosen successor z' via some executable control. We therefore separate (i) *planning objects*—$\hat{\phi}$, a learned QSS critic $Q_\theta(z, z')$, and an estimated reachable set—from (ii) *execution objects*—a mechanism that implements a proposed transition. In our framework the execution mechanism is represented by an implicit inverse controller $\hat{I}(z, z')$ producing a latent control $u \in \mathcal{U}$ intended to drive z to z', together with an interface that converts u into environment actions. The details of when such an $\hat{I}$ exists and is identifiable are deferred to the next section; here we treat it as part of the desired output.

Certified state constraints. Because Reach is unknown, we cannot enforce admissibility directly. Instead, we aim to construct a *certified* reachable set $\widehat{\mathrm{Reach}}_\delta(z) \subseteq \mathcal{Z}_{\mathcal{D}}$ such that edges used for planning are accepted only when a statistical test deems them feasible at confidence level δ. The resulting policy is constrained by construction:

$$\hat{\pi}_{\mathrm{SC}}(z) \in \arg \max_{z' \in \widehat{\mathrm{Reach}}_\delta(z)} Q_\theta(z, z').$$

The key property of certification is control of the false-positive probability $\varepsilon_{\mathrm{fp}}$: the chance that an accepted edge (z, z') is in fact not executable. We also track the false-negative probability $\varepsilon_{\mathrm{fn}}$, which measures the rate at which truly reachable edges are rejected and hence removed from the maximization set.

Target guarantee. Our target is a bound comparing the value of the deployed policy $\hat{\pi}$ to the optimal value under the true state constraint. Under bounded rewards $|r| \le c$ and $\gamma < 1$, we seek a statement of the form

$$V^{\pi^{\star}_{\mathrm{SC}}}(z) - V^{\hat{\pi}}(z) \;\le\; \mathcal{O}\!\left(\frac{\varepsilon_{\mathrm{mdl}} + \varepsilon_{\mathrm{rep}}}{1-\gamma}\right) + \mathcal{O}\!\left(\frac{\varepsilon_{\mathrm{fp}}}{1-\gamma}\right) + \mathcal{O}\!\left(\frac{\varepsilon_{\mathrm{fn}}}{1-\gamma}\right), \qquad \forall z \in \mathcal{Z}_{\mathcal{D}}.$$

Here $\varepsilon_{\mathrm{mdl}}$ captures forward/inverse model inaccuracies on certified edges, and $\varepsilon_{\mathrm{rep}}$ captures deviations of $\hat{\phi}$ from a Markov (or bisimulation-consistent)

representation sufficient for planning. This form isolates the only ways in which action-free planning can fail while remaining within dataset states: either we accept infeasible shortcuts (false positives), we remove valid options (false negatives), or our representation/modeling is inaccurate even on certified transitions. The subsequent sections specify conditions under which each term can be controlled and clarify why some form of identifiability is indispensable for any nontrivial guarantee.

3 Identifiability and Controllability Assumptions

Our algorithmic stance is that, although the environment exposes an action space $\mathcal{A}$, the offline data do not. Hence any deployable policy must be built around objects that can be inferred from pairs (o_t, o_{t+1}), and any executed control must be produced by a mechanism that depends only on the current observation (or latent) and the planned successor. This forces a structural assumption: there must exist a *latent control* variable that is (approximately) determined by the transition itself.

Latent controlled dynamics. We assume that the representation $\phi : \mathcal{O} \to \mathcal{Z}$ induces a controlled latent process admitting a deterministic one-step model

$$z_{t+1} \;=\; \tilde{f}(z_t, u_t), \qquad u_t \in \mathcal{U},$$

where $\mathcal{U}$ is a latent control space that need not coincide with $\mathcal{A}$ (indeed, $\mathcal{U}$ may encode a low-level controller parameterization, a motor primitive index, or an action embedding). Determinism is not a modeling convenience but a means of making the substitution "choose successor state $\rightsquigarrow$ choose control" well-posed: if $\tilde{f}$ is genuinely stochastic in u, then a single desired successor z' does not specify an executable one-step effect, and planning over successor states is no longer equivalent to planning over controls. In practice, we permit residual stochasticity, but it must be absorbed into the error terms (and, crucially, into conservative certification) so that the planning graph remains a faithful abstraction.

One-step controllability on dataset support. Since we restrict planning to $\mathcal{Z}_{\mathcal{D}}$, we do not require global controllability in $\mathcal{Z}$, only that transitions observed in the dataset are explainable by some latent control. Formally, for every transition pair (o, o^+) in $\mathcal{D}$ with $z = \phi(o)$ and $z^+ = \phi(o^+)$, there exists $u \in \mathcal{U}$ such that $\tilde{f}(z, u) = z^+$. This hypothesis ensures that the dataset edges are not merely *correlational* but admit a consistent control-based interpretation in latent space. It is a minimal requirement for any execution mechanism: if an observed successor cannot be realized by any control, then a planner that selects successors from $\mathcal{Z}_{\mathcal{D}}$ may request transitions that are literally infeasible even on in-distribution states.

Identifiability of implicit controls. The critical requirement for learning from action-free data is that the control that realizes a given transition can be inferred from the transition itself. We therefore assume an identifiability condition: for almost every reachable pair (z, z') of interest there is a well-defined inverse map

$$\tilde{I}(z, z') \in \arg\min_{u \in \mathcal{U}} \|\tilde{f}(z, u) - z'\|,$$

and, in the realizable deterministic case, $\tilde{f}(z, \tilde{I}(z, z')) = z'$. We interpret "well-defined" as uniqueness up to a set of pairs of measure zero (or, more generally, that the set of minimizers is sufficiently small that a measurable selection exists). Identifiability is what permits the following operational decomposition:

$$\text{planning: } z \mapsto z' \qquad \rightsquigarrow \qquad \text{execution: } (z, z') \mapsto u = \tilde{I}(z, z'),$$

which is the only route to a deployable controller when action labels are absent.

Two remarks clarify why this is stronger than merely requiring existence of some action that attains z'. First, without identifiability there may be many incompatible controls that produce the same observed transition, and an inverse model trained from state-only data has no anchor for choosing among them. Second, the learned inverse $\hat{I}$ is not intended to recover the true environment action; it must only recover a control variable that is *consistent with* the learned forward model $\hat{f}$ and, through the actuator interface, produces the desired effect in the environment. Identifiability is thus posed at the level of the induced latent control system, not at the level of $\mathcal{A}$.

Examples where identifiability is plausible. Deterministic systems with sufficiently rich actuation often satisfy the above conditions after a suitable choice of ϕ and $\mathcal{U}$. As a stylized example, consider a fully observed deterministic gridworld in which each move deterministically changes the state and for each ordered pair of adjacent states (s, s') there is exactly one primitive action taking s to s'. Then $\mathcal{U}$ may be identified with the set of neighboring displacements, and $\tilde{I}(s, s')$ is simply that displacement. More continuously, in a locally linearizable mechanical system discretized with small time step, one may take u to encode a target velocity increment; then the map $(z, z') \mapsto u$ is often (locally) unique because the state increment over one step determines the required control to first order. In these cases, learning $\hat{I}$ from transitions is akin to learning an inverse kinematics or inverse dynamics map, with the important distinction that the target is a latent control parameter rather than the logged action.

Counterexamples and failure modes. There are two distinct obstructions.

Non-identifiability. Suppose there exist $u_1 \neq u_2$ such that $\tilde{f}(z, u_1) = \tilde{f}(z, u_2)$ on a non-negligible set. Then (z, z') does not determine u, and an inverse model is not learnable from state-only data without additional side information. This ambiguity can be benign if *any* such u is executable and equivalent for future behavior, but it becomes harmful whenever the different controls have different downstream consequences not captured by the one-step transition (e.g., hidden actuator states, contact modes, or delays), in which case the latent process is not Markov in z and the discrepancy is charged to ε_{rep}.

Lack of controllability (spurious edges). Observation-only datasets may contain transitions driven by exogenous variables or by a behavior process that uses capabilities unavailable at deployment. For instance, a dataset collected under human teleoperation may include transitions that an autonomous controller cannot reproduce (due to latency, constraints, or missing sensors). In latent terms, such edges do not belong to the agent-induced reachability relation $\text{Reach}(z)$ even if they appear in $\mathcal{D}$. Any method that treats dataset adjacency as controllability will be systematically optimistic; this is precisely the phenomenon that motivates our insistence on explicit reachability *certification* and the control of false positives.

These counterexamples also underlie the information-theoretic hardness: if two environments generate identical distributions over observation-only datasets but disagree on which successor states are actually achievable, then no algorithm can guarantee near-optimal control uniformly. Consequently, the above identifiability/controllability assumptions are not merely technical; they are the hypotheses that separate the learnable regime from the indistinguishable one.

Implication for our pipeline. Under identifiability, the pair $(\hat{f}, \hat{I})$ becomes a testable hypothesis about executability: if z' is genuinely reachable from z, then applying the inferred control $\hat{I}(z, z')$ and rolling it through $\hat{f}$ should return (approximately) z'. The next section formalizes this intuition by turning prediction residuals into confidence sets $\widehat{\text{Reach}}_\delta(z)$ with calibrated false-positive control, thereby converting the structural assumptions of this section into a concrete retrieve–then–certify reachability graph for state-constrained dynamic programming.

4 Certified Latent Reachability

State-constrained planning requires, for each latent state $z \in \mathcal{Z}_\mathcal{D}$, a set of successor states that are not merely *observed* but are *executable* by the deployed controller. In the action-free setting we cannot test executability by replaying logged actions, nor can we assume that adjacency in $\mathcal{D}$ coincides with the agent-induced reachability relation. We therefore introduce a statistical

object—a *certified* one-step reachability set—that plays the role ordinarily served by the known transition model (or by action labels) in offline RL.

Ideal one-step reachability on dataset support. Let $\mathcal{Z}_{\mathcal{D}} = \{\phi(o_t) : (o_t, o_{t+1}, r_t) \in \mathcal{D}\}$ denote the set of latent states realized in the dataset.[1] The object we would like to plan with is the *true* state-constrained reachability relation

$$\mathrm{Reach}(z) \;:=\; \big\{z' \in \mathcal{Z}_{\mathcal{D}} : \exists u \in \mathcal{U} \text{ s.t. } \tilde{f}(z,u) = z'\big\}.$$

Planning over $\mathrm{Reach}(z)$ ensures that the induced policy never requests a successor z' that cannot be produced in one step by some admissible latent control. However, $\mathrm{Reach}(\cdot)$ is not observed and must be estimated from $\mathcal{D}$.

A model-based executability residual. Given learned models $(\hat{f}, \hat{I})$, a natural test for whether a candidate successor z' is executable from z is whether the inferred control $u = \hat{I}(z, z')$ is consistent with the forward prediction:

$$e(z, z') \;:=\; \big\|\hat{f}\big(z, \hat{I}(z, z')\big) - z'\big\|.$$

Small residual $e(z, z')$ is *necessary* for executability under the learned latent control system: if $\hat{f}$ accurately reflects the true controlled transition and $\hat{I}$ indeed returns a control implementing $z' \mid z$, then $e(z, z')$ should concentrate near 0 for reachable edges. Conversely, if a proposed successor is spurious (e.g., only observed under an exogenous behavior process, or far off the learned controllable manifold), then typically no control produced by $\hat{I}$ will make $\hat{f}$ land near z', yielding a large residual.

Because $(\hat{f}, \hat{I})$ are learned, $e(z, z')$ alone is not a certificate. We require a calibrated decision rule that turns residuals into an acceptance test with explicit control of false positives.

Confidence bounds via ensembles and calibration. We train an ensemble $\{(\hat{f}_j, \hat{I}_j)\}_{j=1}^{E}$ (or an equivalent uncertainty estimator, e.g., dropout) and define an uncertainty-aware residual statistic. One convenient choice is to compute the set of ensemble residuals

$$e_j(z, z') \;:=\; \big\|\hat{f}_j\big(z, \hat{I}_j(z, z')\big) - z'\big\|, \qquad j = 1, \ldots, E,$$

and then form a conservative lower confidence bound (LCB) for the *best-case* one-step prediction error, for instance

$$\mathrm{LCB}(z, z') \;:=\; \mathrm{Quantile}_{\alpha}\big(\{e_j(z, z')\}_{j=1}^{E}\big),$$

where $\alpha \in (0, 1)$ is small (e.g., $\alpha = 0.1$), so that LCB remains small only when the ensemble agrees that the transition is easy to explain. Other monotone

[1] When $\hat{\phi}$ is used in practice, $\mathcal{Z}_{\mathcal{D}}$ refers to the empirical embedding set $\{\hat{\phi}(o_t)\}$.

summaries (mean plus/minus a multiple of standard deviation, or a conformal score built from the full predictive distribution) are admissible; our analysis uses only that $\mathrm{LCB}(z, z')$ is a scalar score whose distribution can be calibrated.

We split the dataset into a training portion and a calibration portion $\mathcal{D}_{\mathrm{cal}}$. From $\mathcal{D}_{\mathrm{cal}}$ we compute calibration scores for observed transitions:

$$s_i \;:=\; \mathrm{LCB}(z_i, z_i^+), \qquad (z_i, z_i^+) \in \phi(\mathcal{D}_{\mathrm{cal}}).$$

We then set a threshold $\tau(\delta)$ as an empirical quantile of $\{s_i\}$ (with the standard finite-sample conformal correction), so that under the usual exchangeability assumptions for calibration residuals, a fresh *in-distribution* reachable transition satisfies

$$\mathbb{P}\big(\mathrm{LCB}(z, z^+) > \tau(\delta)\big) \;\le\; \delta.$$

Operationally, $\tau(\delta)$ is the maximal residual we are willing to treat as "explainable by control" at confidence level $1 - \delta$.

Certified reachability sets and false-positive control. We define the certified reachability set as

$$\widehat{\mathrm{Reach}}_\delta(z) \;:=\; \big\{z' \in \mathcal{Z}_{\mathcal{D}} : \mathrm{LCB}(z, z') \le \tau(\delta)\big\}.$$

The intent is that membership in $\widehat{\mathrm{Reach}}_\delta(z)$ is a high-confidence statement of one-step executability. In particular, if a candidate edge (z, z') is accepted only when $\mathrm{LCB}(z, z') \le \tau(\delta)$, then the probability of accepting a genuinely non-executable edge is controlled by a false-positive rate $\varepsilon_{\mathrm{fp}}$ that can be set (up to finite-sample slack) by δ through the calibration procedure. This is the property required for safe Bellman backups: the critic maximization is restricted to successors whose existence is statistically supported, rather than merely hypothesized by function approximation.

The complementary error is the false-negative rate $\varepsilon_{\mathrm{fn}}$, i.e., rejecting a truly reachable successor. False negatives reduce the option set and can lead to conservatism; they are traded against false positives by δ, the choice of LCB, and the capacity of $(\hat{f}, \hat{I})$. Our bounds account for both effects additively, with false positives being the more structurally dangerous failure mode because they can create optimistic shortcuts in the induced planning graph.

Retrieve–then–certify construction. A direct evaluation of $\widehat{\mathrm{Reach}}_\delta(z)$ against all $z' \in \mathcal{Z}_{\mathcal{D}}$ is prohibitive for large datasets, and unnecessary when controllable one-step successors occupy a small, structured subset. We therefore adopt a two-stage procedure.

First, for each $z \in \mathcal{Z}_{\mathcal{D}}$ we retrieve a small candidate set $C(z) \subset \mathcal{Z}_{\mathcal{D}}$ of size k using an approximate nearest neighbor (ANN) index built on latent embeddings. The retrieval metric is chosen so that nearby latents correspond to plausibly reachable successors (e.g., Euclidean distance in $\mathcal{Z}$ when ϕ is trained with a predictive objective). This step enforces a computational sparsity prior: only candidates "close" to z are even considered.

Second, we certify each candidate $z' \in C(z)$ by computing $\mathrm{LCB}(z, z')$ and applying the calibrated threshold test. The accepted set is then

$$\widehat{\mathrm{Reach}}_{\delta}(z) \;=\; \big\{ z' \in C(z) : \mathrm{LCB}(z, z') \le \tau(\delta) \big\},$$

which is stored as the outgoing adjacency list of z in a sparse directed graph over $\mathcal{Z}_{\mathcal{D}}$.

This retrieve–then–certify pipeline separates concerns: ANN retrieval provides scalability and a weak structural prior, while certification provides the statistical guardrail that prevents the planner from relying on edges unsupported by the learned control system. In the next stage of the method, all Bellman backups and greedy successor selections are restricted to this certified graph, thereby implementing dynamic programming on an induced state-constrained MDP whose transition support is explicitly controlled by δ and the learned model uncertainty.

5 Algorithm: Action-Free State-Constrained QSS Learning (AF-SCQ)

We now describe the learning and planning procedure that instantiates the certified reachability construction of Section 4 into a deployable controller. The guiding principle is to replace action selection by *successor selection* on a certified directed graph over dataset latents, and to realize the selected successor through a learned implicit inverse controller.

Step 1: representation learning. We first learn an encoder $\hat{\phi} : \mathcal{O} \to \mathcal{Z}$ using only observation sequences. Any self-supervised objective that promotes one-step predictability and approximate Markovianity in $\mathcal{Z}$ is admissible; concretely, we may minimize a contrastive forward prediction loss

$$\min_{\phi,\psi} \; \mathbb{E}\Big[\ell\big(\psi(\phi(o_t)), \phi(o_{t+1})\big) \Big],$$

where ψ is a prediction head and ℓ is either an ℓ_2 regression loss or an InfoNCE-style loss with negatives sampled from the replay buffer. We then embed the dataset as $z_i = \hat{\phi}(o_i)$ and $z_i^+ = \hat{\phi}(o_i^+)$, and write $\mathcal{Z}_{\mathcal{D}} = \{z_i\}_{i=1}^N$.

Step 2: learning a latent control system without actions. Because actions are absent, we introduce an *implicit* control variable $u \in \mathcal{U}$ and learn both a forward controlled model $\hat{f} : \mathcal{Z} \times \mathcal{U} \to \mathcal{Z}$ and an inverse controller $\hat{I} : \mathcal{Z} \times \mathcal{Z} \to \mathcal{U}$ so that $\hat{f}(z, \hat{I}(z, z')) \approx z'$ on dataset transitions. A simple and effective training objective is the cycle-consistency regression

$$\min_{f,I} \ \mathbb{E}_{(z,z^+)\sim\phi(\mathcal{D})}\Big[\big\|f\big(z, I(z, z^+)\big) - z^+\big\|^2\Big],$$

optionally augmented with regularization on $u = I(z, z^+)$ (e.g. norm penalties) and with an auxiliary reconstruction/prediction loss to prevent degenerate solutions. In practice we train an ensemble $\{(\hat{f}_j, \hat{I}_j)\}_{j=1}^{E}$ to enable uncertainty quantification for certification. We emphasize that this step does not require a unique "true" action interpretation; it suffices that $\hat{I}$ returns a latent control that is consistently mapped by $\hat{f}$ to the desired successor on the transitions present in $\mathcal{D}$.

Step 3: calibration split and certification threshold. We partition $\mathcal{D} = \mathcal{D}_{\text{train}} \cup \mathcal{D}_{\text{cal}}$. The models $(\hat{f}_j, \hat{I}_j)$ are trained on $\mathcal{D}_{\text{train}}$, and $\mathcal{D}_{\text{cal}}$ is used only to choose the acceptance threshold $\tau(\delta)$ for the ensemble statistic used in Section 4. This separation is essential: the quantity δ is meaningful only insofar as the calibration residuals are exchangeable with test-time residuals for in-distribution transitions.

Step 4: sparse certified reachability graph construction. We build an approximate nearest neighbor (ANN) data structure over $\mathcal{Z}_{\mathcal{D}}$ to support scalable successor proposals. For each anchor state $z \in \mathcal{Z}_{\mathcal{D}}$, we retrieve a candidate set $C(z) \subset \mathcal{Z}_{\mathcal{D}}$ of size k and apply the certification test to each candidate, yielding the adjacency list

$$\widehat{\text{Reach}}_{\delta}(z) \ = \ \big\{z' \in C(z) : \text{LCB}(z, z') \le \tau(\delta)\big\}.$$

We store these sets as a directed graph on $\mathcal{Z}_{\mathcal{D}}$. The role of ANN is purely computational: it restricts attention to a manageable subset of successors; the statistical guardrail is provided only by the certification test.

Step 5: QSS value learning on the certified graph. We learn a critic $Q_\theta(z, z')$ interpreting (z, z') as a *one-step option*: request successor z' and then continue optimally subject to the same constraint. The training data for the critic consists of certified edges (z, z') together with a reward associated to the transition. When rewards are available only for observed dataset pairs, we may either (i) restrict critic updates to certified edges that coincide with observed pairs, or (ii) fit an auxiliary reward model $\hat{r}(z, z')$ on $\phi(\mathcal{D})$ and

use it to score arbitrary certified edges. In either case, we perform fitted Q-iteration with the constrained Bellman target

$$y(z, z') \;=\; \hat{r}(z, z') \;+\; \gamma \max_{\bar{z} \in \widehat{\mathrm{Reach}}_\delta(z')} Q_{\theta^-}(z', \bar{z}),$$

and minimize $\mathbb{E}[(Q_\theta(z, z') - y(z, z'))^2]$ over samples of certified edges, using a target network θ^- for stability. The maximization is always taken over $\widehat{\mathrm{Reach}}_\delta(\cdot)$, which prevents the critic from extrapolating to latent successors unsupported by the certified graph.

Step 6: deployment as successor selection plus inverse realization. At test time, given the current observation o, we compute $z = \hat{\phi}(o)$. Since z need not lie exactly in $\mathcal{Z}_\mathcal{D}$, we optionally "snap" to the nearest dataset embedding $\Pi_{\mathcal{Z}_\mathcal{D}}(z)$ for robustness, or else retrieve and certify successors relative to z on-the-fly using the same ANN and certification statistic. The policy then selects

$$z^\star \in \arg \max_{z' \in \widehat{\mathrm{Reach}}_\delta(z)} Q_\theta(z, z'),$$

and outputs the latent control command $u = \hat{I}(z, z^\star)$. The environment-facing action is produced by an actuator interface that implements u (e.g. by mapping u to joint torques, or by feeding u as a goal to a lower-level controller). Under the identifiability assumptions stated in the enclosing scope, this realizes the intended one-step transition up to the controlled error accounted for by certification.

Summary of invariants enforced by construction. By design, (i) all planning decisions select successors inside a subset of dataset states, so multi-step rollouts remain on-support to the extent permitted by $\widehat{\mathrm{Reach}}_\delta$; and (ii) each one-step request is filtered by a calibrated executability test, ensuring that the edges used by the Bellman backups and by the deployed greedy policy satisfy an explicit false-positive control determined by δ and the calibration procedure. These invariants are precisely those required for the contraction-based analysis in the next section, where we view AF-SCQ as approximate dynamic programming on the induced state-constrained MDP defined by the certified graph.

6 Theory I: Upper Bounds via Contraction and Certified Policy Evaluation

We now analyze AF-SCQ as approximate dynamic programming on a state-constrained control problem induced by the dataset latents. Throughout, we restrict attention to latent states in $\mathcal{Z}_\mathcal{D}$ and assume bounded rewards $|r| \le c$ and discount $\gamma \in [0, 1)$.

The induced state-constrained MDP and QSS parameterization. Given the true latent dynamics $\tilde{f}$ and the realizability/identifiability assumptions in the enclosing scope, we may treat *successor selection* as the effective decision. Let the true one-step feasible successor set be

$$\mathrm{Reach}(z) \;:=\; \mathcal{Z}_{\mathcal{D}} \cap \mathrm{SR}(z),$$

where $\mathrm{SR}(z)$ denotes the one-step reachable set under the environment controls (equivalently, under latent controls $u \in \mathcal{U}$). A state-constrained policy π_{SC} selects a successor $z' \in \mathrm{Reach}(z)$ and then realizes it via the inverse controller; thus we may define the (optimal) constrained value as

$$V^{\pi_{\mathrm{SC}}}(z) \;=\; \mathbb{E}\Big[\sum_{t\geq 0} \gamma^t r(z_t, z_{t+1}) \;\Big|\; z_0 = z,\; z_{t+1} \sim P(\cdot \mid z_t, z_{t+1} \in \mathrm{Reach}(z_t))\Big].$$

The QSS critic $Q(z, z')$ assigns value to the one-step option "request successor z' and continue optimally under the same constraint."

True and certified Bellman optimality operators. Define the true state-constrained Bellman optimality operator $\mathcal{T}^*$ acting on bounded functions $Q : \mathcal{Z}_{\mathcal{D}} \times \mathcal{Z}_{\mathcal{D}} \to \mathbb{R}$ by

$$(\mathcal{T}^* Q)(z, z') \;:=\; r(z, z') \;+\; \gamma \max_{\bar{z} \in \mathrm{Reach}(z')} Q(z', \bar{z}), \qquad z' \in \mathrm{Reach}(z), \quad (1)$$

and (for convenience) leave $(\mathcal{T}^* Q)(z, z')$ undefined or arbitrary when $z' \notin \mathrm{Reach}(z)$ since such pairs are never selected by an admissible policy.

Analogously, the algorithm operates on the *certified* reachable sets $\widehat{\mathrm{Reach}}_\delta(z) \subseteq \mathcal{Z}_{\mathcal{D}}$ and a learned reward model $\hat{r}$ (or rewards restricted to observed pairs). This yields the induced operator

$$(\widehat{\mathcal{T}} Q)(z, z') \;:=\; \hat{r}(z, z') \;+\; \gamma \max_{\bar{z} \in \widehat{\mathrm{Reach}}_\delta(z')} Q(z', \bar{z}), \qquad z' \in \widehat{\mathrm{Reach}}_\delta(z). \quad (2)$$

The fitted Q-iteration updates of AF-SCQ are precisely approximate applications of $\widehat{\mathcal{T}}$.

Contraction. Both operators are γ-contractions in the ℓ_∞ norm on their respective domains of definition.

Lemma 6.1 (Contraction on feasible edges)**.** *Let Q_1, Q_2 be bounded. Then for any pair (z, z') with $z' \in \mathrm{Reach}(z)$,*

$$\big|(\mathcal{T}^* Q_1)(z, z') - (\mathcal{T}^* Q_2)(z, z')\big| \;\leq\; \gamma \|Q_1 - Q_2\|_\infty,$$

and for any pair (z, z') with $z' \in \widehat{\mathrm{Reach}}_\delta(z)$,

$$\big|(\widehat{\mathcal{T}} Q_1)(z, z') - (\widehat{\mathcal{T}} Q_2)(z, z')\big| \;\leq\; \gamma \|Q_1 - Q_2\|_\infty.$$

Consequently, each operator admits a unique fixed point Q^* and $\widehat{Q}^*$ on its induced graph, and repeated exact application converges geometrically.

The proof is immediate from the non-expansiveness of the max operator and the fact that rewards do not depend on Q.

From approximate Bellman updates to policy loss. Given any candidate Q, define the corresponding constrained greedy successor-selection rule

$$\pi_Q(z) \in \arg \max_{z' \in \widehat{\mathrm{Reach}}_\delta(z)} Q(z, z').$$

We compare the deployed policy $\hat{\pi} = \pi_{\widehat{Q}}$ to the optimal true constrained policy $\pi^\star_{\mathrm{SC}}$ (which is greedy w.r.t. Q^* on the true graph). Standard approximate dynamic programming arguments reduce performance loss to (i) the error in evaluating the induced operator and (ii) the mismatch between the true and certified operators.

To make this precise, let us measure the *operator mismatch* on pairs that the algorithm may consider:

$$\varepsilon_{\mathrm{op}} := \sup_{z \in \mathcal{Z}_{\mathcal{D}}} \sup_{z' \in \widehat{\mathrm{Reach}}_\delta(z)} \left| (\widehat{\mathcal{T}} Q^*)(z, z') - (\mathcal{T}^* Q^*)(z, z') \right|. \tag{3}$$

In addition, let $\varepsilon_{\mathrm{approx}}$ denote the residual with which we solve the fixed point of $\widehat{\mathcal{T}}$ (capturing function approximation and optimization error), e.g.

$$\varepsilon_{\mathrm{approx}} \geq \|\widehat{Q} - \widehat{\mathcal{T}} \widehat{Q}\|_\infty.$$

A simulation-lemma style argument (or the usual bound for greedy policies under approximate Q-functions) yields the qualitative implication

$$V^{\pi^\star_{\mathrm{SC}}}(z) - V^{\hat{\pi}}(z) \lesssim \frac{1}{1-\gamma} \big(\varepsilon_{\mathrm{op}} + \varepsilon_{\mathrm{approx}} \big), \qquad \forall z \in \mathcal{Z}_{\mathcal{D}}, \tag{4}$$

where the constant depends only on the choice of norm and whether we translate QSS values into state values via $V(z) = \max_{z'} Q(z, z')$.

Decomposing operator mismatch into FP/FN and modeling errors. We now bound $\varepsilon_{\mathrm{op}}$ in terms of three contributions.

First, *model and reward error* on executable edges: on any truly reachable (z, z'), the learned quantities $(\hat{f}, \hat{I}, \hat{r})$ may induce an error in one-step prediction and immediate reward. We collect these terms into $\varepsilon_{\mathrm{mdl}}$, which upper bounds the induced discrepancy between $\hat{r}(z, z')$ and $r(z, z')$ as well as the effect of imperfect inverse realization on the next latent state, when the edge is indeed executable.

Second, *representation error*: even if $\hat{f}$ is accurate in $\mathcal{Z}$, the latent representation may fail to be perfectly Markov/bisimulation-preserving. Such failures appear exactly as an additional operator perturbation (e.g. the conditional distribution of z_{t+1} depends on history beyond z_t), which we summarize by $\varepsilon_{\mathrm{rep}}$.

Third, *reachability errors*: the certified graph may contain false positives (FP) and omit true edges (FN). We treat FP edges as the more dangerous case because the maximization in (2) can propagate spurious optimistic shortcuts. Let $\varepsilon_{\mathrm{fp}}$ be an upper bound on the probability (over the algorithmic certification randomness and the calibration guarantee) that an accepted edge is in fact not executable in the true latent system; similarly let $\varepsilon_{\mathrm{fn}}$ bound the probability that a truly executable candidate is rejected. Under conformal/ensemble calibration (Section 4), we have $\varepsilon_{\mathrm{fp}} \le \delta$ up to finite-sample slack.

To translate these into value loss, we note that any single spurious edge can change a max backup by at most a bounded amount. Since $|r| \le c$, the range of discounted returns is at most $2c/(1-\gamma)$, and hence the impact of selecting a wrong successor in a backup is controlled by a constant $\Delta = \mathcal{O}\big(\frac{c}{1-\gamma}\big)$. It follows that, in operator norm,

$$\varepsilon_{\mathrm{op}} \;\le\; \varepsilon_{\mathrm{mdl}} + \varepsilon_{\mathrm{rep}} \;+\; \varepsilon_{\mathrm{fp}} \cdot \Delta \;+\; \mathcal{O}\Big(\varepsilon_{\mathrm{fn}} \cdot \Delta\Big), \tag{5}$$

where the FN term reflects that removing feasible options can only decrease the max backup, producing a pessimistic bias whose magnitude is again bounded by Δ.

Performance bound. Combining (4) and (5) yields the stated form of our upper bound.

Theorem 6.2 (Certified reachability performance bound)**.** *Assume bounded rewards $|r| \le c$, $\gamma < 1$, and that the certification procedure ensures false-positive rate at most $\varepsilon_{\mathrm{fp}}$ and false-negative rate at most $\varepsilon_{\mathrm{fn}}$. Let $\widehat{Q}$ be the output critic with Bellman residual $\varepsilon_{\mathrm{approx}}$. Then the deployed greedy policy $\hat{\pi}$ satisfies, for all $z \in \mathcal{Z}_{\mathcal{D}}$,*

$$V^{\pi^\star_{\mathrm{SC}}}(z) - V^{\hat{\pi}}(z) \;\le\; \frac{2}{1-\gamma}\big(\varepsilon_{\mathrm{mdl}} + \varepsilon_{\mathrm{rep}} + \varepsilon_{\mathrm{approx}}\big) \;+\; \mathcal{O}\Big(\frac{\varepsilon_{\mathrm{fp}}}{1-\gamma}\Big) \;+\; \mathcal{O}\Big(\frac{\varepsilon_{\mathrm{fn}}}{1-\gamma}\Big).$$

This theorem formalizes the role of certification: the only way the maximization in the Bellman backups can become arbitrarily wrong is through false positives, and the calibration parameter δ directly upper bounds their probability. The remaining terms are intrinsic approximation errors stemming from learning $\hat{\phi}$, $(\hat{f}, \hat{I})$, and (optionally) $\hat{r}$, as well as the optimization error in fitting $\widehat{Q}$.

7 Theory II: Lower Bounds and an Information-Theoretic Impossibility

We now justify the identifiability condition in the enclosing scope by exhibiting a worst-case obstruction: when we observe only (o_t, o_{t+1}, r_t) (equiv-

alently, only latent pairs (z_t, z_{t+1}, r_t) after representation) and do not observe nor intervene on actions, the control structure of the environment is not, in general, identifiable. In particular, there may exist distinct MDPs whose *observation-only* data distributions coincide, yet whose optimal state-constrained policies (hence optimal constrained values) differ by a constant. Any algorithm mapping such datasets to a deployable policy must therefore incur constant suboptimality on at least one of the indistinguishable environments.

The core issue is that the dataset identifies only a *mixture* of controlled transition kernels. If the behavior process generating $\mathcal{D}$ follows an unknown (possibly non-stationary) action selection rule $\mu(a \mid s)$, then the induced marginal transition is

$$P_\mu(s' \mid s) \;=\; \sum_{a \in \mathcal{A}} \mu(a \mid s)\, P(s' \mid s, a).$$

From state-only trajectories, we can estimate P_μ (and the associated reward conditional on observed transitions), but we cannot generally recover the family $\{P(\cdot \mid s, a)\}_{a \in \mathcal{A}}$, nor even decide whether a particular observed transition is *controllable* or merely occurs stochastically under a single action. In the language of AF-SCQ, the learner may infer that an edge (z, z') is feasible because it appears in $\mathcal{D}$, but the existence of that edge in passive data does not imply the existence of a control u with $\tilde{f}(z, u) = z'$.

Theorem 7.1 (Impossibility without identifiability)**.** *Fix any (possibly randomized) learning algorithm that takes as input an offline dataset $\mathcal{D} = \{(o_i, o_i^+, r_i)\}_{i=1}^N$ (with no actions) and outputs a deployable policy $\hat{\pi}$ (possibly via learned latent models and an implicit inverse controller). Then there exist two MDPs $\mathcal{M}_1, \mathcal{M}_2$ and a data-generating behavior process such that:*

1. *the induced distributions over datasets coincide, i.e. $\mathcal{D} \sim \mathbb{P}_1$ under $\mathcal{M}_1$ and $\mathcal{D} \sim \mathbb{P}_2$ under $\mathcal{M}_2$ satisfy $\mathbb{P}_1 = \mathbb{P}_2$ on (o, o^+, r),*

2. *yet the optimal state-constrained values differ, and consequently the algorithm suffers constant worst-case suboptimality:*

$$\inf_{alg} \sup_{\mathcal{M} \in \{\mathcal{M}_1, \mathcal{M}_2\}} \mathbb{E}\big[V^{\pi^\star_{\mathrm{SC}}}(z_0) - V^{\hat{\pi}}(z_0)\big] \;\geq\; \Omega(1),$$

   for a designated start latent state $z_0 \in \mathcal{Z}_\mathcal{D}$.

Proof sketch. We present an explicit indistinguishability construction on a three-state system. Let $\mathcal{S} = \{s_0, s_G, s_B\}$ with s_G, s_B absorbing. Let observations be fully revealing (so we may take $o = s$ and ϕ as the identity), and define rewards by $r(s_0, s_G) = 1$, $r(s_0, s_B) = 0$, and $r(\cdot, \cdot) = 0$ otherwise. Consider the following two MDPs.

Environment $\mathcal{M}_1$ (controllable). Let $\mathcal{A} = \{a_G, a_B\}$. Dynamics from s_0 are deterministic:

$$P(s_G \mid s_0, a_G) = 1, \qquad P(s_B \mid s_0, a_B) = 1,$$

and s_G, s_B are absorbing under both actions.

Environment $\mathcal{M}_2$ (uncontrollable). Let $\mathcal{A} = \{\bar{a}\}$ be a singleton. Dynamics from s_0 are stochastic:

$$P(s_G \mid s_0, \bar{a}) = \tfrac{1}{2}, \qquad P(s_B \mid s_0, \bar{a}) = \tfrac{1}{2},$$

and s_G, s_B are absorbing.

Now define the behavior process generating the offline data. In $\mathcal{M}_1$, the behavior policy chooses a_G and a_B with equal probability $\frac{1}{2}$ at s_0; in $\mathcal{M}_2$, there is only $\bar{a}$. Under these choices, the *marginal* distribution of one-step transitions from s_0 is identical in the two environments: with probability $\frac{1}{2}$ the next state is s_G (yielding reward 1) and with probability $\frac{1}{2}$ it is s_B (reward 0). Since the remainder of each trajectory is absorbing with zero reward, the induced distribution over datasets of tuples (o, o^+, r) is exactly the same under $\mathcal{M}_1$ and $\mathcal{M}_2$ for any sample size N.

However, the optimal control values differ. In $\mathcal{M}_1$, the optimal (state-constrained or unconstrained) policy selects a_G at s_0 and obtains value $V^{\pi^\star}(s_0) = 1$ (the episode ends immediately with reward 1). In $\mathcal{M}_2$, no policy can influence the transition and thus $V^{\pi^\star}(s_0) = \frac{1}{2}$. The gap is therefore $\frac{1}{2}$, independent of N and of γ (since rewards occur only at the first step).

Because the dataset laws coincide, any algorithm must output the *same* distribution over policies when run on $\mathcal{D}$ from $\mathcal{M}_1$ as when run on $\mathcal{D}$ from $\mathcal{M}_2$. Consequently, if we evaluate the output policy in the true environment, it cannot be simultaneously near-optimal in both: the sum of regrets across the two environments is at least the value gap, hence the worst-case regret is at least half the gap, i.e. at least $\frac{1}{4} = \Omega(1)$. This establishes the claim. □

Several remarks connect this lower bound to our setting. First, the construction persists under latent modeling: even if we learn a perfect predictor for the *behavior-induced* transition law (here, "from s_0 go to s_G with probability $\frac{1}{2}$"), such a model does not specify the set $\mathrm{Reach}(s_0)$ of *controllable* successors. In $\mathcal{M}_1$, s_G is reachable by a specific action (and thus should be an admissible successor under a state-constrained successor-selection policy), whereas in $\mathcal{M}_2$ it is not reliably realizable. Any implicit inverse controller $\hat{I}(s_0, s_G)$ is therefore underdetermined by passive data: it may exist in $\mathcal{M}_1$ and be meaningless in $\mathcal{M}_2$, despite the datasets being identical.

Second, the graph viewpoint makes the obstruction more vivid. The dataset reveals an undirected adjacency between s_0 and $\{s_G, s_B\}$ in the

sense that both transitions appear. What is unidentifiable is the *action-labeled directionality*—whether there exists a control that deterministically (or with high probability) selects s_G as the successor. The maximization in any Bellman backup that treats observed transitions as selectable edges is precisely what fails in $\mathcal{M}_2$.

Finally, the lower bound is information-theoretic rather than computational: it does not rely on function approximation, finite samples, or optimization difficulties. It states that, absent an identifiability condition (or additional side information such as actions, interventions, or experimentally varied controls), no amount of offline observation-only data suffices to guarantee vanishing suboptimality in the worst case. Having established this necessity, we turn next to computational questions: given identifiability and a certifiable reachability test, how do we scale successor retrieval, certification, and QSS backups to large N without quadratic enumeration over state pairs.

8 Computational Complexity: Scalability via Approximate Retrieval and Certification

We analyze the computational and memory costs incurred by AF-SCQ when instantiated at dataset scale. The dominant difficulty is the construction and subsequent use of the certified successor relation $z' \in \widehat{\mathrm{Reach}}_\delta(z)$ without enumerating all pairs $(z, z') \in \mathcal{Z}_\mathcal{D}^2$. A naive quadratic construction would (i) propose every z' as a candidate successor for each z, (ii) run the reachability test for each pair, and (iii) perform Bellman maximizations over all dataset states; this yields time and space costs on the order of $\Theta(N^2)$, which is intractable even before accounting for ensemble-based uncertainty.

Representation and latent-model training. Let $d_z = \dim(\mathcal{Z})$ and let the representation encoder $\hat{\phi}$ and latent dynamics components $(\hat{f}, \hat{I})$ be parameterized by networks of total size W. Training costs depend on the optimizer and architecture; we therefore track the cost at the level of forward/backward passes. Under minibatch SGD for T gradient steps with batch size B, the training time is $\tilde{O}(T \cdot B \cdot \mathrm{cost}(W))$, with memory $\tilde{O}(W)$ for parameters plus $O(B \cdot d_z)$ for activations (up to standard checkpointing). This stage is typically linear (up to log factors) in the amount of data processed and does not introduce any explicit dependence on N^2.

When we use ensembles to obtain a lower confidence bound (LCB) for one-step prediction error, the parameter footprint scales by the ensemble size E (either explicitly, as E independent networks, or implicitly, via dropout-style approximations). In the explicit case, the training-time multiplier is approximately E if models are trained independently, though in practice we may share the encoder $\hat{\phi}$ and train only heads, reducing the constant factor.

Importantly, even with ensembles, training remains a pass over $\mathcal{D}$ rather than a pass over $\mathcal{Z}_{\mathcal{D}}^2$.

ANN-based candidate successor retrieval. The reachability graph construction begins by embedding each observation o_i to $z_i = \hat{\phi}(o_i)$ and building an approximate nearest neighbor (ANN) index over $\mathcal{Z}_{\mathcal{D}} = \{z_i\}_{i=1}^N$. With index structures such as HNSW or IVF-based indices, the build time is typically $\tilde{O}(N)$ to $\tilde{O}(N \log N)$, with memory $O(N \cdot d_z)$ for stored vectors plus an index-dependent overhead that is usually near-linear in N. In exchange, per-query retrieval of k candidate next states $C(z) = \{z'_1, \ldots, z'_k\}$ costs $O(\log N + k)$ to $O(\log N + k \log k)$ (again index-dependent), rather than $O(N)$ for brute-force search. Performing retrieval for all dataset states thus scales as

$$\tilde{O}\big(N(\log N + k)\big),$$

which is quasi-linear for fixed k. The parameter k directly controls the maximum out-degree of the induced sparse graph and therefore controls both planning cost and memory.

We note that ANN retrieval introduces an additional approximation beyond statistical error: even if a truly reachable successor z' is near z in the embedding, the index may fail to return it. This phenomenon effectively increases false negatives in $\widehat{\mathrm{Reach}}_\delta$ and should be regarded as contributing to $\varepsilon_{\mathrm{fn}}$ (or to an algorithmic analogue thereof), with the usual tradeoff: more aggressive retrieval (larger k, higher recall settings) reduces missed candidates but increases certification cost.

Certification cost and calibration overhead. For each candidate pair (z, z') with $z' \in C(z)$, the algorithm evaluates the residual

$$e(z, z') \;=\; \big\|\hat{f}\big(z, \hat{I}(z, z')\big) - z'\big\|,$$

and accepts the edge if a calibrated LCB for $e(z, z')$ is below a threshold $\tau(\delta)$. With an explicit ensemble of size E, a typical LCB computation requires E forward evaluations of $\hat{f}$ (and, depending on implementation, either E evaluations of $\hat{I}$ or a shared $\hat{I}$). Hence certification for one candidate pair costs $O(E)$ network evaluations, and certification over all retrieved pairs costs

$$O\big(N \cdot k \cdot E\big) \quad \text{(network-evaluation units)}.$$

This is the central computational term specific to reachability construction, and it should be contrasted with the quadratic baseline $O(N^2 E)$ that would arise from certifying all pairs.

Calibration using conformal prediction (or any held-out quantile scheme) adds a one-time cost. If we reserve N_{cal} transitions, then we must compute

calibration residuals (again at $O(E)$ evaluations per residual) and then compute a quantile. The quantile computation itself is $O(N_{\text{cal}})$ after residuals are obtained; thus calibration costs $O(N_{\text{cal}}E)$ model evaluations and negligible additional memory. This overhead is linear and does not alter asymptotic scalability.

Memory footprint of the certified reachability graph. After certification, we store for each z the accepted set $\widehat{\text{Reach}}_\delta(z) \subseteq C(z)$. Let $d(z) = |\widehat{\text{Reach}}_\delta(z)|$ and $\bar{d} = \frac{1}{N}\sum_z d(z)$. Since $d(z) \le k$, we store at most Nk directed edges. The memory is therefore

$$O\big(N \cdot d_z\big) \;+\; O(N \cdot \bar{d})$$

for embeddings plus adjacency (storing indices and optionally per-edge metadata such as predicted reward or residual statistics). This should be compared to the $O(N^2)$ adjacency matrix implied by quadratic enumeration. In large-scale regimes, adjacency storage is often dominated by the embeddings and the ANN index rather than by the sparse edge list.

QSS critic training and backup complexity. Training the critic $Q_\theta(z, z')$ uses Bellman targets of the form

$$\hat{r}(z, z') + \gamma \max_{\bar{z} \in \widehat{\text{Reach}}_\delta(z')} Q_{\theta^-}(z', \bar{z}).$$

The inner maximization ranges over $\widehat{\text{Reach}}_\delta(z')$, which has size $d(z') \le k$, not over all N states. If we perform M SGD updates for the critic and evaluate Q_{θ^-} on all successors of z' in each update, the total target-construction cost is $O(M \cdot k)$ forward passes through the critic (up to batching). This sparsity is precisely the mechanism by which we avoid the unconstrained "max over $\mathcal{Z}$" that would induce both computational blowup and out-of-distribution extrapolation.

Deployment-time complexity. At test time, given an observation o, we compute $z = \hat{\phi}(o)$, retrieve candidates $C(z)$ via ANN, certify (or reuse cached certification if z is a dataset state), select $z^\star \in \widehat{\text{Reach}}_\delta(z)$ maximizing $Q_\theta(z, z^\star)$, and output the latent control $\hat{I}(z, z^\star)$. If we certify on the fly, the per-step cost is $O(\log N + kE + k)$ model evaluations; if we cache $\widehat{\text{Reach}}_\delta(z)$ for dataset states and use nearest-neighbor projection from z to a nearby dataset latent, the cost reduces to $O(\log N + k)$. In either case, the dependence on N is polylogarithmic rather than linear.

Tradeoffs and comparison to quadratic enumeration. Quadratic enumeration requires considering every potential successor, yielding time $\Theta(N^2E)$ for certification and $\Theta(N^2)$ for storing or repeatedly recomputing adjacency, which is prohibitive. Approximate retrieval reduces candidate generation to $\tilde{O}(N(\log N + k))$ and limits certification to $O(NkE)$, while the critic uses max operations of cost $O(k)$ rather than $O(N)$. The price is that k and ANN recall become algorithmic parameters that mediate a three-way tradeoff among (i) computational budget, (ii) conservatism via false negatives (missing truly reachable successors), and (iii) statistical safety via certification (controlling false positives). In the regimes we target, choosing k so that Nk fits in memory and NkE fits in compute is the appropriate scaling principle; the resulting graph is sparse by design, and all subsequent dynamic programming operations inherit this sparsity.

9 Experiments: Evidence for Action-Free State-Constrained Control

We evaluate AF-SCQ in a *withheld-actions* setting: when an offline benchmark provides action logs, we discard actions during training and use only tuples (o_t, o_{t+1}, r_t) as prescribed by the problem definition. Actions are used only at evaluation time to (i) execute the environment with the control output produced by our deployed inverse controller, and (ii) compute diagnostic quantities (e.g., whether a proposed transition was in fact achievable under the true dynamics). This protocol isolates the contribution of certified reachability and implicit inverse control while allowing comparison to standard action-logged offline RL methods.

Benchmarks and modalities. We consider three families of datasets. (1) **D4RL state-based tasks** (e.g., locomotion and manipulation), where o_t is a low-dimensional state vector. This setting isolates action-free learning from representation learning and tests the identifiability assumptions most directly. (2) **Pixel variants** in which o_t is an image observation rendered from the same underlying tasks; here the learned representation $\hat{\phi}$ is necessary, and representation error ε_{rep} becomes visible. (3) **Video-only datasets** in which the observations are short clips (stacked frames) with sparse or delayed rewards; in this case we treat o_t as a temporal window and train $\hat{\phi}$ with a sequence encoder (e.g., a small ConvNet+GRU), so that $z_t = \hat{\phi}(o_{t-h:t})$ is approximately Markov.

Methods compared. Our primary comparison is to state-constrained Q-learning with actions available (SCQL), which serves as an upper bound on what can be achieved when the same constraint class is used but the true actions are known. We also report standard offline RL baselines requiring

actions (e.g., CQL/IQL-style methods), included to contextualize the cost of action-free learning rather than as strictly comparable methods under the stated constraints. For action-free baselines, we include (i) a one-step model selection heuristic that chooses z' by nearest-neighbor reward prediction without Bellman backup, and (ii) an ablation that removes certification and accepts all ANN-proposed candidates (denoted NoCert), which tests the necessity of controlling false positives.

Training and evaluation protocol. For all tasks we train $\hat{\phi}$ and the latent models $(\hat{f}, \hat{I})$ on $\mathcal{D}$ only. We reserve a calibration split $\mathcal{D}_{\text{cal}}$ to set $\tau(\delta)$ for a target confidence level $\delta \in \{0.05, 0.1\}$. We then construct $\widehat{\text{Reach}}_\delta$ and train the QSS critic $Q_\theta(z, z')$ using only certified edges. At deployment, given the current observation o, we compute $z = \hat{\phi}(o)$, retrieve candidate successors, select $z^\star \in \widehat{\text{Reach}}_\delta(z)$ maximizing $Q_\theta(z, z^\star)$, and execute the implied control $u = \hat{I}(z, z^\star)$. When the underlying environment expects primitive actions $a \in \mathcal{A}$, we instantiate an actuator interface by training a small action-decoder h on (o_t, o_{t+1}) pairs (still without using logged actions) to output continuous commands; in settings where a simulator provides a known low-level controller, we use it as the interface and treat u as its command input.

We report (i) normalized return on D4RL tasks, (ii) a *constraint violation proxy* defined as the fraction of executed transitions whose realized next latent state $\hat{\phi}(o_{t+1})$ falls outside a fixed-radius neighborhood of $\mathcal{Z}_\mathcal{D}$ (measured by ANN distance), and (iii) a *reachability precision* diagnostic computed in simulator-based tasks: among accepted edges (z, z'), we estimate the empirical false-positive rate by attempting to realize z' from z via the deployed controller and declaring success if the achieved next state is within tolerance.

Withheld-actions D4RL results. Across state-based D4RL tasks, AF-SCQ consistently improves upon action-free nearest-neighbor heuristics and remains competitive with action-logged state-constrained methods when identifiability is approximately satisfied (deterministic or low-noise dynamics and sufficiently rich coverage in $\mathcal{D}$). The gap to SCQL is smallest in environments where one-step transitions are nearly single-modal and where ANN retrieval yields high recall of true successors; in these cases the certified graph $\widehat{\text{Reach}}_\delta$ recovers a large fraction of the effective support of the behavior policy while filtering spurious edges, yielding stable Bellman backups. Conversely, in tasks with pronounced stochasticity or contact-induced discontinuities, we observe conservative graphs (high estimated ε_{fn}) and correspondingly reduced returns, consistent with the qualitative dependence predicted by the bound in Theorem 2.

Pixel and video results. On pixel variants, the dominant sensitivity is representation quality. Using a contrastive predictive objective for $\hat{\phi}$ (tem-

poral InfoNCE with data augmentations) yields substantially better downstream value estimation than purely reconstructive autoencoders, which tend to preserve nuisance variation and degrade neighborhood structure for retrieval. In video-only datasets, treating o_t as a short window improves the Markov property in z_t and reduces compounding error in one-step certification; without temporal context, the residuals $e(z, z')$ become multi-modal and calibration tightens $\tau(\delta)$, removing too many edges. Empirically, we find that certification remains beneficial in these settings: NoCert can obtain higher apparent values under the learned critic but suffers from frequent execution failures (large constraint violation proxy), indicating reliance on false-positive shortcuts.

Ablations: calibration, retrieval size, representation. We ablate (i) calibration strategy, (ii) retrieval budget k, and (iii) representation family. Replacing conformal/quantile calibration with an uncalibrated fixed threshold produces unstable behavior across datasets: in some tasks the graph becomes overly dense and yields catastrophic optimistic planning, while in others it collapses to near-empty reachability, both of which degrade returns. Varying k exhibits the expected tradeoff: small k limits planning depth through reduced branching and increases effective false negatives; large k improves recall but increases the number of marginal candidates, which either increases certification cost or forces a looser threshold (raising false positives). In our experiments, moderate k achieves the best return–violation tradeoff, and we observe a monotone relationship between tighter δ (smaller ε_{fp}) and lower violation rates.

Representation ablations confirm that metrics induced by $\hat{\phi}$ are not interchangeable: we require that local neighborhoods in $\mathcal{Z}$ correspond to dynamically plausible successors. Predictive representations (contrastive or forward-model-based) dominate static embeddings; moreover, adding an auxiliary bisimulation-style penalty (matching reward and predicted next-latent distributions) improves stability in tasks with perceptual aliasing, suggesting a concrete mechanism for reducing ε_{rep}.

Failure cases and diagnostics. We observe three recurrent failure modes. First, when identifiability fails (e.g., two distinct controls produce indistinguishable (o_t, o_{t+1}) transitions under the dataset distribution but differ off-distribution), $\hat{I}$ can be systematically wrong while still fitting one-step prediction, leading to execution-time mismatch not detected by purely latent residuals. Second, when the dataset has strong coverage gaps, ANN retrieval returns candidates that are near in embedding but separated by unmodeled constraints; certification may reject these but then yields disconnected graphs, producing myopic policies. Third, in highly stochastic dynamics, the residual distribution becomes heavy-tailed and calibration tightens, ef-

fectively converting stochastic reachability into false negatives. In all cases, the principal practical diagnostic is to inspect the degree distribution of $\widehat{\mathrm{Reach}}_\delta$ and the execution-time violation proxy; both correlate with performance and provide actionable signals for adjusting δ, k, and representation learning.

These experiments support the central thesis: when actions are unavailable, planning over *certified* dataset transitions can recover substantial control performance while explicitly controlling the failure mode induced by false-positive reachability edges. The remaining limitations align with the impossibility phenomenon of Theorem 4, motivating the connections to prior work discussed next.

10 Related Work and Discussion

Offline RL and distributional constraints. Offline RL has largely been studied in the action-logged setting, where one estimates action-conditioned values and must control extrapolation error induced by maximizing over unseen actions. Representative approaches include conservative objectives that penalize out-of-distribution (OOD) actions (e.g., CQL-style penalties), implicit behavior regularization (e.g., IQL-style expectile regression), and model-based variants that restrict rollouts to the dataset support. Our setting differs at the level of information: we do not observe actions, hence classical action-level OOD detection is not well-posed. The state-constrained viewpoint we adopt is therefore closer to the "support-constrained MDP" perspective in which planning is restricted to empirically supported transitions. In particular, our use of QSS-style backups over pairs (z, z') parallels state-constrained Q-learning formulations in which the maximization set is a reachable neighborhood of dataset states, thereby avoiding unconstrained maximization over a continuous action space. The principal distinction is that we must additionally *realize* selected successors via an implicit inverse controller, which introduces an identifiability requirement that is absent when actions are logged.

Action-free learning and control from state-only trajectories. There is a long history of extracting control signals from observations without explicit action labels, including system identification under unknown inputs, inverse optimal control, and methods that infer "control" variables as latent causes of state transitions. In RL-adjacent communities, this appears as control from observations, imitation from state-only demonstrations, and behavior modeling where the policy is implicit in the transitions. Our formulation makes explicit a separation between (i) estimating an *induced reachability relation* over dataset latents and (ii) learning an inverse mapping $\hat{I}(z, z')$ that implements chosen transitions. This separation clarifies why

action-free learning is not merely a matter of replacing $Q(s, a)$ by $Q(s, s')$: without an identifiability condition ensuring that the chosen successor can be produced by a well-defined control variable, the planning objective is ambiguous, consistent with the impossibility phenomenon formalized earlier. In this sense, our state-only algorithm can be viewed as approximate dynamic programming on a reduced control system whose inputs are the successor states themselves, with $\hat{I}$ serving as the actuator interface.

World models, video prediction, and visual planning. Model-based control from pixels has been studied extensively, often via predictive latent variable models and planning in latent space. A typical pipeline learns an encoder and a dynamics model, then performs model predictive control (MPC) by searching over action sequences to maximize a reward model or a goal likelihood. In contrast, we do not assume access to actions for either learning or planning; consequently, classical latent-MPC methods that require sampling action sequences are not directly applicable. Our use of successor-state selection is closer to planning on a graph induced by observed transitions, but with two substantive additions: (a) we require a certification step to control false-positive edges that would otherwise create optimistic shortcuts, and (b) we incorporate an explicit inverse map $\hat{I}(z, z')$ so that selected successors correspond to executable controls. From the perspective of video-based control, $\hat{I}$ plays a role analogous to "visual servoing" or goal-conditioned control modules, except that the "goal" is a nearby dataset latent state chosen by a Bellman backup rather than a user-specified target.

Latent action models and identifiability. Learning latent actions (or "action embeddings") from observation transitions is closely related to unsupervised skill discovery, option learning, and structured latent variable models in which a discrete or continuous latent u is inferred to explain multimodal transition distributions. Many such approaches are designed for representation or generative modeling, and they often permit non-identifiable parameterizations: multiple latent encodings can fit the same transition distribution. For control, however, non-identifiability is not a benign modeling choice; it manifests as a failure to produce reliable execution when the policy selects a successor state that the inferred u cannot realize in the true environment. Our assumption that $\tilde{I}(z, z')$ is well-defined (up to measure-zero ambiguities) can be interpreted as requiring that the latent control variable corresponds to a genuine controllable degree of freedom rather than an arbitrary mixture component. This aligns with classical identifiability concerns in inverse dynamics and with causal perspectives in which interventions (controls) must be distinguishable from passive correlations. Practically, it motivates architectures and training criteria for $\hat{I}$ that encourage functional invertibility (e.g., consistency losses between $\hat{f}$ and $\hat{I}$) and evaluation proto-

cols that test execution fidelity, not merely one-step prediction error.

Representation learning, bisimulation, and metric structure. Our algorithm relies on nearest-neighbor retrieval in $\mathcal{Z}$ to propose candidate successors, so the induced metric structure of $\hat{\phi}$ is not incidental: neighborhoods must reflect dynamical plausibility rather than purely perceptual similarity. This connects directly to bisimulation and representation learning for RL, where one seeks embeddings in which states with similar reward and transition structure are close. Bisimulation metrics and related objectives provide a principled route to controlling ε_{rep} by enforcing that distances in $\mathcal{Z}$ upper bound behavioral dissimilarity. Contrastive predictive objectives (e.g., temporal discrimination) can be viewed as approximating such structure by making consecutive (or model-predicted) latents closer than negatives, thereby improving ANN recall for true successors. Unlike work that uses representation learning primarily to accelerate value function approximation, we use it to define the *candidate set* over which the Bellman operator is applied; thus, representation error affects not only approximation quality but also the feasible action set of the induced constrained MDP.

Uncertainty, certification, and safe deployment. Conformal prediction, ensembles, and other uncertainty quantification tools have been used for model-based RL to produce calibrated prediction sets, safe MPC, and robust planning. We employ a closely related idea but apply it to *edge acceptance* in a dataset-induced reachability graph. This use is structurally similar to safe exploration methods that restrict actions to those with certified outcomes, except that we operate entirely offline and the restriction is over dataset successors rather than primitive actions. Conceptually, the certification step is our analog of conservative action selection: it curtails optimistic value propagation by limiting Bellman backups to edges whose one-step realization error is controlled. The resulting guarantees are necessarily stated in terms of false positives and false negatives in the induced graph; this is appropriate because, in the action-free setting, reachability itself is a learned object rather than a given property of the environment.

Implications for robotics and autonomy. Robotic systems often possess heterogeneous control stacks (high-level commands, learned low-level controllers, safety filters), and logs may omit the exact motor torques or may be collected from sources where action semantics differ (teleoperation, shared autonomy, or legacy controllers). Our framework targets precisely this mismatch: it plans over states (or observations) that are actually seen in the log and delegates realization to an inverse controller appropriate for the platform. The state-constrained and certified nature of the planner suggests a practical deployment strategy in which one prefers conservative,

high-confidence transitions among familiar states, thereby reducing the likelihood of entering unmodeled regimes. At the same time, our impossibility result indicates a clear boundary: without some form of identifiability (architectural, physical, or via auxiliary supervision), action-free logs cannot determine which transitions are controllable, so any autonomous deployment must either assume such structure or incorporate additional sensing/actuation signals. We view this as a constructive message: when action channels are missing or unreliable, one should invest in (i) representations that preserve controllability-relevant geometry, and (ii) calibrated certification mechanisms that make the induced constraints explicit and auditable.

11 Conclusion

We studied offline control from observation-only transition data, in which the dataset has the form $\mathcal{D} = \{(o_t, o_{t+1}, r_t)\}$ and no action labels are available. The central difficulty in this setting is not merely statistical but informational: without additional structure, observation transitions do not determine what the agent can *cause*, hence planning objectives that implicitly assume controllability of arbitrary observed successors are ill-posed. Our approach makes this issue explicit by (i) planning in a *state-constrained* latent space over dataset-supported successors and (ii) separating *selection* of a desired successor latent state from *realization* of that successor via a learned inverse controller.

Concretely, we introduced AF-SCQ, which learns an encoder $\hat{\phi} : \mathcal{O} \to \mathcal{Z}$, a latent forward model $\hat{f}$, and an inverse controller $\hat{I}(z, z')$ intended to output a latent control $u \in \mathcal{U}$ that realizes the transition $z \to z'$. Using an ANN structure over $\mathcal{Z}_{\mathcal{D}}$, we propose candidate successors and certify one-step reachability by accepting edges (z, z') only when a calibrated uncertainty criterion indicates that the realized prediction error is small, yielding a certified neighborhood $\widehat{\mathrm{Reach}}_{\delta}(z) \subseteq \mathcal{Z}_{\mathcal{D}}$. We then perform QSS-style backups on pairs (z, z'), where all maximizations are restricted to $\widehat{\mathrm{Reach}}_{\delta}(\cdot)$, thereby preventing unconstrained value extrapolation over latent space and ensuring that the deployed policy $\hat{\pi}$ selects only certified dataset successors. Under bounded rewards and standard realizability/calibration conditions, this construction yields an approximate dynamic programming procedure on a certified state-constrained MDP induced by $\widehat{\mathrm{Reach}}_{\delta}$, with performance degradation controlled additively by false-positive/false-negative certification rates and by model/representation errors.

At the theoretical level, our results delineate a sharp boundary. With an identifiability condition—informally, that for dataset transitions there exists a well-defined inverse mapping $\tilde{I}(z, z')$ that corresponds to a genuine controllable degree of freedom—action-free planning reduces to state-constrained Q-learning on successor states, since actions serve only to instantiate selected

successors. Conversely, without identifiability, we cannot in general distinguish passive correlations from controllable transitions, and we established an impossibility phenomenon: there exist environments that induce identical observation-only datasets but admit different optimal state-constrained policies, implying worst-case constant suboptimality for any action-free algorithm. We view this pair of statements as the appropriate conceptual framing of action-free offline RL: what can be achieved hinges on which aspects of controllability are identifiable from the available signals and inductive biases.

Several limitations remain. First, our certification mechanism is fundamentally one-step. While one-step false-positive control is already sufficient to prevent the most direct optimistic shortcuts in Bellman backups, it does not automatically yield strong guarantees about multi-step execution, because compounding errors and distribution shift along policy-induced trajectories may cause the realized roll-out to deviate from the graph edges that were certified locally. One next step is therefore *multi-step reachability certification*: for instance, certifying not only that z' is reachable from z in one step, but that a short-horizon plan $z \to z_1 \to \cdots \to z_H$ is executable with controlled failure probability. Technically, this suggests combining per-edge certificates with compositional bounds (e.g., union bounds or martingale-style concentration) or learning a tube-like invariant set around $\mathcal{Z}_{\mathcal{D}}$ in which the closed-loop inverse controller remains accurate. A complementary direction is to certify *robust reachability*, where acceptance depends on a worst-case realization error over an uncertainty set for $(\hat{f}, \hat{I})$, leading to graph edges that are conservative by design.

Second, our analysis relies on an induced Markov property in latent space and treats representation error as an additive term. In realistic partially observed settings, even a strong encoder $\hat{\phi}$ may fail to be sufficient for control, so planning over $z = \hat{\phi}(o)$ may conflate aliased histories. A natural extension is to incorporate memory explicitly, e.g., by replacing $\phi(o_t)$ with $\phi(h_t)$ for a recurrent state estimator over histories $h_t = (o_{t-\ell:t})$, or by moving from an MDP reduction to a belief-state or predictive-state representation. In such settings, the object we constrain to the dataset support may no longer be a set of instantaneous latents but a set of *filtered* latents, and the reachability certification must account for estimator uncertainty as well as dynamics uncertainty. More generally, identifying conditions under which action-free data yield a controllable representation in POMDPs—and clarifying the minimal auxiliary signals needed when they do not—remains open.

Third, our framework plans over dataset states, which is both a strength (conservatism) and a limitation (coverage). When the optimal constrained policy requires transitions that are not present in $\mathcal{D}$, no amount of offline computation can recover them. This motivates principled data acquisition strategies and hybrid pipelines in which offline learning produces a conservative controller that can be safely deployed, while subsequent interaction (if permitted) is used to expand $\mathcal{Z}_{\mathcal{D}}$ in targeted ways. Even without training-

time interaction, one may attempt limited generalization by certifying neighborhoods around dataset states rather than only the states themselves; doing so without reintroducing OOD value extrapolation is subtle and likely requires stronger regularity assumptions on $\tilde{f}$ and on the encoder geometry.

Finally, multi-agent and non-stationary settings present an additional identifiability challenge: state-only logs may reflect the actions of other agents or changes in the environment, so the same observed transition (o_t, o_{t+1}) may not be reproducible by the learner. Extending the reachability notion to explicitly include latent variables representing other agents (or regimes) suggests either a latent-mixture certification scheme or a causal formulation in which we seek transitions stable under intervention by the learner. Here the impossibility boundary becomes even more salient: without some separation of "self" versus "other" causes, action-free control is not merely hard; it is undefined.

In summary, we presented a state-constrained, certified approach to action-free offline control that is explicit about what is being planned (successor dataset states) and how it is executed (via an inverse controller), together with guarantees that are naturally expressed in terms of reachability certification errors and modeling/representation inaccuracies. The overarching message is methodological as much as algorithmic: in the absence of action labels, the correct conservative object is not an action set but a *reachable successor set* that must be learned, certified, and kept auditable.