TCGI: TRANSITION-CONSISTENCY-GUIDED IMAGINATION FOR VISUAL IMITATION LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

World models support visual imitation learning by generating latent trajectories for policy optimization. Expert demonstrations provide transition information that can also guide the construction of these trajectories. We introduce Transition-Consistency-Guided Imagination (TCGI), which incorporates expert transition preferences into conservative model-based imitation learning (CMIL). A learned scorer distinguishes demonstrated latent transitions from unfiltered world-model rollouts. Its scores weight the fusion of candidate successor states, augment the imitation reward, and control when an ensemble-disagreement penalty is applied. These uses connect demonstration supervision to imagined state construction and policy optimization within a fixed imagination horizon. At deployment, the actor operates directly on inferred latent states. We evaluate TCGI on manipulation and locomotion tasks, with component ablations and parameter sensitivity analyses. Reported performance gains vary across tasks and metrics, with trade-offs between late-training scores and performance over recorded evaluation intervals. TCGI provides a concrete mechanism for incorporating expert transition preferences into latent imagination and a basis for examining how this guidance affects visual imitation learning.

1 INTRODUCTION

Visual imitation learning uses demonstrations to acquire control policies without requiring a task reward for training. Behavior cloning fits demonstrated actions (Pomerleau, 1988), but policy errors can lead to states poorly covered by the demonstrations (Ross et al., 2011). World models offer a complementary source of experience: an agent can learn from the consequences of its actions in a compact latent dynamics model (Hafner et al., 2020). Combining world models with imitation objectives supports visual policy learning through imagined rollouts (Rafailov et al., 2021b; Kolev et al., 2024).

Demonstrations can guide the construction of these rollouts as well as their reward. In conservative model-based imitation learning (CMIL; Kolev et al., 2024), a state action discriminator supplies an imitation reward, and an ensemble penalty discourages exploitation of uncertain predictions. The demonstrated successor provides another signal: it can inform preferences among candidate transitions generated for the same state and action. This motivates learning expert transition preferences at the point where the world model constructs the next imagined state.

We propose *Transition-Consistency-Guided Imagination* (TCGI) within the CMIL framework. An ensemble of transition scorers distinguishes expert posterior transitions from unfiltered policy imagination. TCGI uses the scores to weight candidate successor states, add a reward bonus, and condition the disagreement penalty. These uses connect the same learned preference to imagined state construction and the policy objective. TCGI retains CMIL's reconstruction-based world model, actor critic backbone, fixed imagination horizon, and direct actor execution.

Transition scoring also underlies reward-consistent model learning in offline reinforcement learning (Luo et al., 2024). TCGI learns preferences over expert posterior and policy-imagined latent transitions, then fuses candidate state fields; MOREC selects a successor. The score expresses a preference relative to these training distributions. This formulation places the focus on how expert transition information is used to construct imagined experience.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

We evaluate TCGI on eight simulated tasks spanning Franka Kitchen, ShadowHand Baoding Balls, MetaWorld, and DeepMind Control. The reported comparisons show higher mean Last-10 return and return AUC than CMIL on ShadowHand, small mean differences on Kitchen, and task-dependent gains on the additional suites. Component ablations and parameter sweeps compare performance summaries across guidance configurations.

Our contributions are:

- A transition-guided imagination procedure within CMIL that uses expert transition scores for field-wise candidate fusion, reward augmentation, and a conditional disagreement penalty, with fixed-horizon training and direct policy execution.
- An empirical analysis across eight simulated tasks, with component ablations and hyperparameter sensitivity that characterize task- and metric-dependent behavior.

2 Related Work

Imitation objectives and behavioral regularization. Behavior cloning (BC) fits demonstrated actions by supervised learning (Pomerleau, 1988; Ross et al., 2011). Inverse reinforcement learning instead infers a learning objective from expert behavior, through learned costs (Finn et al., 2016), adversarial distribution matching (Ho & Ermon, 2016; Fu et al., 2018), or inverse soft-Q learning (Garg et al., 2021). These objectives admit a common divergence-minimization perspective (Ghasemipour et al., 2020). Discriminator-Actor-Critic improves interaction efficiency with off-policy updates (Kostrikov et al., 2019). Optimal-transport approaches construct imitation rewards by matching learner and expert distributions (Dadashi et al., 2021); regularized optimal transport (ROT) combines such rewards with BC pretraining and adaptive behavioral regularization (Haldar et al., 2023). TCGI retains BC regularization and adds a transition score that shapes the imagined experience used for policy optimization.

Visual imitation with world models. PlaNet learns recurrent latent dynamics for planning, while Dreamer and its successors train policies through latent imagination (Hafner et al., 2019; 2020; 2021; 2025). Visual Adversarial Imitation Learning using Variational Models (V-MAIL) combines visual representation learning with adversarial imitation on model-generated rollouts (Rafailov et al., 2021b). CMIL extends this direction with conservative policy optimization in a world model (Kolev et al., 2024), building on latent offline policy optimization and offline-to-online fine-tuning (Rafailov et al., 2021a; 2023). Other approaches match latent trajectories offline (DeMoss et al., 2025), couple discrimination with model and policy learning (Zhang et al., 2023), or separate task-relevant dynamics from visual distractors (Wan et al., 2023). Recent methods use reward-free world models with inverse soft-Q learning and planning (Li et al., 2025), or compositional digital twins to generate imitation data (Barcellona et al., 2025). TCGI builds on CMIL's reconstruction-based model and actor critic learning, adding expert-guided candidate fusion to imagined training trajectories.

Conservatism and transition guidance. Model-based policy optimization must balance synthetic experience against model bias (Janner et al., 2019). Offline methods address this problem through uncertainty penalties (Yu et al., 2020b), pessimistic dynamics (Kidambi et al., 2020; Rigter et al., 2022), conservative value learning (Kumar et al., 2020; Yu et al., 2021), and model Bellman inconsistency (Sun et al., 2023). CLARE applies conservatism to model-based reward learning (Yue et al., 2023). MOREC learns a dynamics reward from real and adversarially generated transitions, then uses it for candidate selection and rollout termination (Luo et al., 2024). TCGI learns from expert posterior and policy-imagined latent transitions, fuses candidate state fields, and retains fixed-horizon imagination. Its world model is trained by reconstruction and latent regularization. These choices distinguish its scoring distribution, model objective, and rollout operator from MOREC. Appendix A develops the connections to representation learning, transport rewards, and uncertainty-aware model use.

3 Problem Setup and Preliminaries

We consider a partially observed control problem with hidden environment state x_t, visual observation o_t, continuous action a_t, transition kernel $P(x_{t+1} \mid x_t, a_t)$, and discount factor $\gamma \in (0, 1)$. The

learner receives a fixed expert dataset $\mathcal{D}_E = \{(o_0, a_0, o_1, \ldots)\}$ and collects additional interaction data in replay buffer $\mathcal{B}$. Environment rewards and success flags measure evaluation performance; the training reward is learned from demonstrations.

The inherited recurrent state-space model (RSSM; Hafner et al., 2019) represents observation history with deterministic memory h_t and stochastic latent state s_t:

$$h_{t+1} = f_\theta(h_t, s_t, a_t), \qquad s_{t+1} \sim p_\theta(\cdot \mid h_{t+1}), \qquad s_t \sim q_\theta(\cdot \mid h_t, e_\theta(o_t)). \tag{1}$$

Here θ denotes world-model parameters, e_θ the image encoder, p_θ the latent prior, and q_θ the observation-conditioned posterior. The policy feature is $z_t = [s_t, h_t]$. We reserve S_t for the full RSSM state record, which additionally stores distribution parameters and ensemble statistics; $z_t = \mathrm{feat}(S_t)$. The actor is $\pi_\psi(a_t \mid z_t)$, with parameters ψ. The world model learns image reconstruction and a balanced posterior prior Kullback Leibler (KL) loss from replay that includes expert and online experience. Appendix B specifies the architecture and losses.

CMIL supplies the state action imitation objective. A Bernoulli discriminator $D_\omega(z, a) \in (0, 1)$ distinguishes expert state action pairs from unfiltered policy imagination. Its negative log-likelihood is

$$\mathcal{L}_D = -\mathbb{E}_{(z,a)\sim\mathcal{D}_E} \log D_\omega(z, a) - \mathbb{E}_{(z,a)\sim\mathcal{I}_\pi} \log(1 - D_\omega(z, a)), \tag{2}$$

where $\mathcal{I}_\pi$ denotes unfiltered rollouts rooted in replay posterior states. Expert observations are encoded by the current world model. Gaussian input perturbations regularize discriminator training. The inherited implementation uses the Bernoulli mean $D_\omega(z, a)$ as the imitation reward, whereas CMIL's paper uses log-odds. Two action-value networks and a behavior cloning regularizer provide the inherited policy-learning components.

4 Transition-Consistency-Guided Imagination

TCGI connects expert transition preferences to imagined state construction and policy optimization. Figure 1 shows the shared scorer and its three uses: candidate fusion, a score bonus, and a conditional disagreement penalty.

4.1 Learning Expert Transition Preferences

Including the successor lets the scorer distinguish candidate outcomes for a shared state and action. Each head $r_{\phi_j}(z, a, z') \in \mathbb{R}$ maps the current feature, action, and successor feature to a logit. The J trainable heads have parameters ϕ_j. Positive tuples come from adjacent expert posterior states; negative tuples come from the current actor's unfiltered world-model rollouts. Each head minimizes

$$\mathcal{L}_r(\phi_j) = \mathbb{E}_{(z,a,z')\sim\mathcal{D}_E} \mathrm{softplus}[-r_{\phi_j}(z, a, z')] + \mathbb{E}_{(z,a,z')\sim\mathcal{I}_\pi} \mathrm{softplus}[r_{\phi_j}(z, a, z')], \tag{3}$$

where $\mathrm{softplus}(v) = \log(1 + \exp v)$. Training perturbs both endpoint features with independent Gaussian noise and updates only active heads; actions are unperturbed.

For rollout scoring, we average the logits of the active heads and a finite queue of frozen historical ensembles. If $\mathcal{H}$ is the set of available heads,

$$\bar{r}_\phi(z, a, z') = \frac{1}{|\mathcal{H}|} \sum_{j\in\mathcal{H}} r_{\phi_j}(z, a, z'). \tag{4}$$

Unfiltered negatives define the scoring objective before guidance is applied. The score measures compatibility with expert tuples, not calibrated transition accuracy; Appendix B.5 details its distributional interpretation.

4.2 Generating and Fusing Candidate States

Weighted fusion constructs one successor while retaining contributions from multiple scored candidates. For an imagined state S_t, we sample $a_t \sim \pi_\psi(\cdot \mid \mathrm{sg}(z_t))$, where sg denotes stop-gradient, and make K stochastic calls to the RSSM transition, all conditioned on the same (S_t, a_t). These

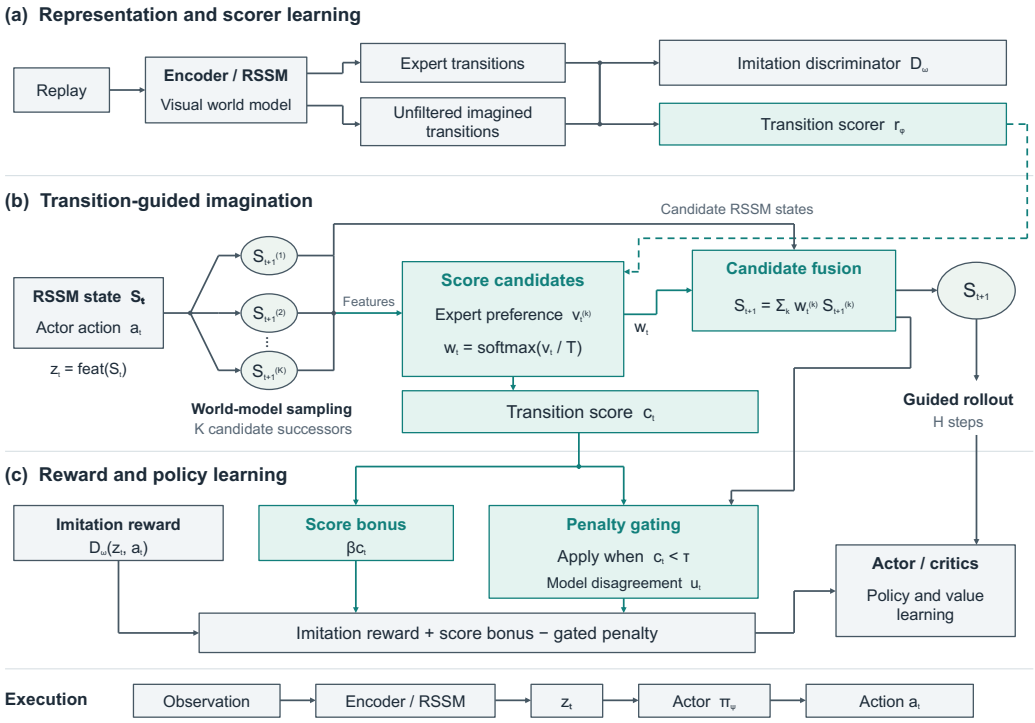


Figure 1: TCGI overview. Expert and unfiltered imagined transitions train a transition scorer. Its scores guide candidate-state fusion, provide a reward bonus, and gate the model-disagreement penalty for policy learning. The bottom path shows direct actor execution.

calls produce candidate records $S_{t+1}^{(k)}$ and features $z_{t+1}^{(k)} = \text{feat}(S_{t+1}^{(k)})$. The scores and normalized weights are

$$v_t^{(k)} = \bar{r}_\phi(\text{sg}(z_t), a_t, \text{sg}(z_{t+1}^{(k)})), \qquad w_t^{(k)} = \frac{\exp(v_t^{(k)}/T)}{\sum_{\ell=1}^{K} \exp(v_t^{(\ell)}/T)}, \tag{5}$$

where $T > 0$ controls weight concentration. For every tensor field b in the RSSM state record, TCGI constructs

$$S_{t+1}[b] = \sum_{k=1}^{K} w_t^{(k)} S_{t+1}^{(k)}[b], \qquad c_t = \sum_{k=1}^{K} \text{sg}(w_t^{(k)}) v_t^{(k)}. \tag{6}$$

Fusion combines sampled states, recurrent memory, Gaussian parameters, and prior-head statistics with the same weights, then recomputes the features. The next RSSM step uses the fused stochastic state and recurrent memory. This is a latent interpolation: the fused stochastic state is not generally a sample from the Gaussian specified by the averaged mean and standard deviation. The scalar c_t summarizes candidate scores without re-scoring the fused transition.

State fusion retains gradients through candidate tensors and weights; c_t detaches only the weights. Scorer endpoint features are detached, while actions remain differentiable, allowing actor gradients through a fixed scorer. Smaller T concentrates the weights. The world-model reconstruction objective is unchanged.

4.3 Reward Augmentation and Conditional Penalties

The world model has M prior heads. For stochastic latent dimension d_s, let $\mu_{t+1,m,i}$ be head m's predicted mean for coordinate i, as stored in the successor record. The disagreement statistic uses the sample SD across heads:

$$u_t = \frac{1}{d_s} \sum_{i=1}^{d_s} \text{SD}_{m=1,\ldots,M}(\mu_{t+1,m,i}). \tag{7}$$

The score bonus rewards expert-preferred candidates; the conditional penalty retains conservatism where their weighted score is low. TCGI combines them as

$$\widetilde{r}_t = D_\omega(z_t, a_t) + \beta c_t - \alpha \mathbf{1}\{\mathrm{sg}(c_t) < \tau\} u_t, \qquad \alpha > 0, \quad \beta \geq 0. \tag{8}$$

Here β weights the score bonus and τ gates the disagreement cost. Sharing the score links candidate construction to the learning objective. This conditional disagreement penalty, called the *safety gate* in the implementation, is a training heuristic. Imagination retains fixed horizon H and discount γ.

Fusion uses relative scores; the bonus and gate use their weighted level. Because softmax weights sum to one even for low-scoring candidates, the bonus provides a complementary expert-compatibility signal. For fixed (S_t, a_t), candidates share the recurrent update and full prior-head predictions, so fusion leaves current-step disagreement u_t unchanged. Guidance affects the gate through c_t and can change future disagreement through subsequent states and actions. The gate changes only the reward, preserving the horizon and discount.

4.4 Policy Optimization and Execution

After updating the world model and both discriminators, freeze their parameters and generate a guided rollout for actor learning. Define $Q_t = \min_{i=1,2} Q_{\eta_i}(z_t, a_t)$ for critic parameters η_1, η_2 and $V_t^\lambda = (1-\lambda)Q_t + \lambda G_t^\lambda$ with trace parameter λ, where G_t^λ is the bootstrapped return specified in Appendix B.3. The actor loss is

$$\mathcal{L}_\pi = -\mathbb{E}_t\big[\gamma^t\big(V_t^\lambda + \eta_{\mathrm{ent}}\,\mathrm{Ent}(\pi_\psi(\cdot \mid \mathrm{sg}(z_t)))\big)\big] + \eta_{\mathrm{BC}}\mathcal{L}_{\mathrm{BC}}, \tag{9}$$

where Ent denotes policy entropy and $\mathcal{L}_{\mathrm{BC}}$ is expert action negative log-likelihood. The expectation averages rollout batches and $t = 0, \ldots, H-2$; η_{ent} and η_{BC} weight regularization. The mixture of Q_t and G_t^λ adds an interpolation outside the λ-return recursion, as in the inherited actor objective. With $H = 15$, training generates 15 transitions and uses the first 14 rewards, with Q_{14} as the bootstrap; the final transition reward is unused.

Rollouts start from detached replay posteriors. Actor learning updates only actor parameters, retaining input derivatives through the fixed world model, discriminator, scorer, and critics. Critics then fit detached targets from replay and a fresh guided rollout. Imagined targets use $\widetilde{r}_t$; replay targets use D_ω for the observed step and bootstrap into guided continuation at its successor. The resulting joint regression is a surrogate objective with different immediate rewards for the two data sources; Appendix B.3 gives both losses.

At execution, the encoder RSSM infers the latent state and the actor outputs an action directly. Candidate generation, transition scoring, and reward augmentation are confined to training. The deployed policy therefore uses the same observation and action interface as CMIL, without running the candidate search at each environment step. Algorithm 1 summarizes training; Appendix B.1 gives the exact update order and gradient paths.

5 Experiments

We evaluate TCGI across eight simulated tasks through baseline comparisons, component removals, and guidance-parameter sweeps. All policies are trained separately for each task.

5.1 Experimental Setup

Tasks and observations. We evaluate Franka Kitchen (Gupta et al., 2020), ShadowHand Baoding Balls (Nagabandi et al., 2020; Kolev et al., 2024), three MetaWorld manipulation tasks (Yu et al., 2020a), and three DeepMind Control Suite (DMC) locomotion tasks (Tunyasuvunakool et al., 2020). Kitchen requires completing four subtasks and reports full-task success. Baoding Balls tests coordinated in-hand manipulation in the PDDM environment. Its episode return sums rewards that penalize ball drops, deviations from moving ball targets, and excessive wrist elevation, capturing ball retention and motion-tracking quality. MetaWorld reports task success; DMC reports episode return. Model-based agents use 64×64 red, green, and blue (RGB) observations and expert observation action sequences. Appendix C specifies the environments and demonstrations.

Algorithm 1 TCGI: transition-guided imagination for visual imitation

Require: Expert data $\mathcal{D}_E$; horizon H; candidates K; T, β, τ, α

```
Initialize the world model, actor, critics, discriminator, and scorer ensembles.
Seed replay B with demonstrations; apply the warm-up in Appendix B.1.
while the interaction budget remains do
    Execute the actor in the environment and add observations/actions to B.
    for each scheduled learner update do
        Sample replay/expert sequences; update the world model (Eq. (10)).
        Generate unfiltered policy rollouts from detached replay posterior states.
        Update D_ω and active scorers (Eqs. (2), (3)); refresh history periodically.
        Start a guided rollout at each replay posterior state.
        for t = 0, ..., H − 1 do
            Sample a_t and K RSSM successors conditioned on (S_t, a_t).
            Score candidates and normalize weights (Eqs. (4), (5)).
            Fuse successors and form c_t (Eq. (6)); compute r̃_t (Eqs. (7), (8)).
        end for
        Update the actor with the guided rollout and expert actions (Eq. (9)).
        Update critics using a fresh guided rollout and replay; synchronize targets periodically.
    end for
end while
return The encoder RSSM actor policy; execution requires no candidate search.
```

Baselines and implementation. Behavior cloning (BC; Pomerleau, 1988) fits expert actions by supervised learning. ROT (Haldar et al., 2023) combines an optimal-transport imitation signal with behavioral regularization. CMIL (Kolev et al., 2024) is the direct model-based baseline, sharing the RSSM and actor critic framework. Kitchen and ShadowHand compare all four methods; the additional suites compare CMIL and TCGI. The archived BC and ROT implementations include task adapters, with numerical stabilization of ROT's transport solver. TCGI uses $K = 8$ candidates, temperature $T = 1$, bonus weight $\beta = 0.1$, penalty threshold $\tau = -1$, disagreement coefficient $\alpha = 10$, and imagination horizon $H = 15$. Appendix B specifies networks and optimizers.

Training budgets and computational cost. Experiments used NVIDIA GeForce RTX 4090 or RTX 5090 GPUs for neural-network computation and CPUs for MuJoCo physics simulation (Todorov et al., 2012). Principal experiments had nominal budgets of 250,000 steps: optimization updates for offline BC and method-specific interaction steps for online methods. These counters do not specify matched environment-interaction costs. The estimated aggregate wall-clock runtime across all experimental configurations was approximately 9,300 hours, summed over individual runs, including those executed in parallel.

Metrics and aggregation. Last-10 averages the final ten evaluation-round means; Best is the largest round mean. Area under the learning curve (AUC) is normalized by each recorded evaluation interval, retaining task-metric units. Intervals are not extrapolated to a common origin. Returns are compared within tasks. All reported metrics are higher-is-better. We report means and sample standard deviations (SDs) of per-seed summaries. All experiments use four training seeds per configuration, except sensitivity settings, which use three. The CMIL and TCGI main-experiment references in the sensitivity table use four seeds. Boldface marks the largest reported means, not statistical significance. Appendices E and F provide individual seed values for the main robotic comparisons, ablations, and sensitivity settings.

5.2 Main Results

Late-training and peak performance on Baoding Balls. TCGI's mean Last-10 return was 31.4 reward units higher than CMIL's (-112.2 versus -143.6), and its recorded-interval AUC was 14.3 higher (-208.2 versus -222.5; Table 1). BC had the highest Best score, -7.8, alongside a Last-10 return of -366.5. Thus, the peak-round and late-training summaries ranked the methods differently: TCGI led Last-10 and AUC, whereas BC led Best. The aggregate return combines ball retention, target tracking, and wrist elevation; these summaries do not separate their contributions.

Table 1: Robotic manipulation: mean ± sample SD across four training seeds. Kitchen reports success (%); ShadowHand reports episode return. AUC uses the corresponding task metric.

Method	Last-10	Best	AUC
Franka Kitchen: Success (%)			
BC	36.8±10.5	70.0±8.2	32.6±6.1
ROT	82.5±25.8	**100.0±0.0**	56.2±25.2
CMIL	96.0±2.0	**100.0±0.0**	75.7±1.4
TCGI (ours)	**96.8±3.4**	**100.0±0.0**	**76.3±1.7**
ShadowHand Baoding Balls: Return			
BC	-366.5±92.0	**-7.8±0.9**	-377.6±33.7
ROT	-516.8±3.5	-13.5±2.3	-448.7±68.4
CMIL	-143.6±24.8	-10.4±2.4	-222.5±43.5
TCGI (ours)	**-112.2±23.5**	-9.4±1.3	**-208.2±31.1**

Table 2: Cross-task evaluation: mean ± sample SD. MetaWorld reports success (%); DMC reports return at 250k agent steps. AUC uses the corresponding task metric. All results are averaged over four training seeds per method and task. Bold indicates the highest mean within each task and metric.

Task	Method	Last-10	Best	AUC
MetaWorld: Success (%)				
Hammer	CMIL	83.0±4.4	**100.0±0.0**	61.7±4.0
	TCGI (ours)	**95.0±0.8**	**100.0±0.0**	**80.5±3.0**
Dial-turn	CMIL	5.5±3.7	27.5±5.0	3.5±1.1
	TCGI (ours)	**11.5±4.8**	**45.0±10.0**	**9.0±2.0**
Sweep-into	CMIL	3.5±3.5	15.0±7.1	2.1±1.2
	TCGI (ours)	**7.0±2.8**	**25.0±7.1**	**4.7±1.6**
DeepMind Control Suite: Return				
Walker Walk	CMIL	911.1±8.3	931.5±0.5	821.7±0.2
	TCGI (ours)	**916.2±8.8**	**936.7±2.7**	**830.2±21.7**
Walker Run	CMIL	656.0±12.5	672.4±0.5	437.8±3.4
	TCGI (ours)	**672.0±0.6**	**681.7±2.6**	**519.2±12.1**
Cheetah Run	CMIL	645.6±0.9	652.7±3.1	578.9±7.2
	TCGI (ours)	**649.7±0.2**	**656.3±0.9**	**592.5±7.2**

A small difference on Kitchen. TCGI's mean Last-10 success was 96.8%, compared with CMIL's 96.0%; their AUCs were 76.3% and 75.7%. These differences were small relative to between-seed variability. Both methods and ROT reached 100% Best success, saturating that metric. Kitchen therefore showed high success with little separation between TCGI and CMIL. The larger gaps from BC and ROT compare different learning frameworks as well as different guidance mechanisms.

Results across manipulation and locomotion. Table 2 covers six additional tasks. In MetaWorld, TCGI's mean Last-10 success exceeded CMIL's by 12.0 percentage points on Hammer, 6.0 on Dial-turn, and 3.5 on Sweep-into. Absolute success remained low on Dial-turn and Sweep-into, at 11.5% and 7.0% for TCGI. In DMC, the Last-10 mean differences were 5.1 on Walker Walk, 16.0 on Walker Run, and 4.1 on Cheetah Run. Walker Run also had the largest recorded-interval AUC difference in this suite, 81.4. Across these separately trained policies, TCGI had higher Last-10 and

Table 3: Component ablations: mean ± sample SD across four seeds per row. Kitchen reports success (%); ShadowHand reports return.

Variant	Last-10	Best	AUC
Franka Kitchen			
CMIL	96.0±2.0	**100.0±0.0**	75.7±1.4
w/o Filter	93.8±4.4	**100.0±0.0**	73.9±0.9
w/o Bonus	96.5±1.0	**100.0±0.0**	73.3±4.1
w/o Safety	95.0±4.3	**100.0±0.0**	74.3±4.4
TCGI	**96.8±3.4**	**100.0±0.0**	**76.3±1.7**
ShadowHand Baoding Balls			
CMIL	-143.6±24.8	-10.4±2.4	-222.5±43.5
w/o Filter	-242.4±121.7	-11.8±1.8	-218.4±39.0
w/o Bonus	-181.5±59.0	-10.8±2.1	**-204.8±22.3**
w/o Safety	-154.8±79.7	-10.3±1.0	-223.2±17.5
TCGI	**-112.2±23.5**	**-9.4±1.3**	-208.2±31.1

AUC means on all six tasks, with substantial variation in the size of the differences and the absolute task performance.

5.3 Component Ablations

All removal variants retain scorer training. *w/o Filter* uses a single model successor and replaces the weighted score c_t with that successor's score, which still supplies the bonus and gate. *w/o Bonus* sets $\beta = 0$. *w/o Safety* removes the disagreement penalty entirely, so it compares the gated penalty with zero penalty, not with CMIL's unconditional penalty. These variants compare coupled configurations; the single-candidate contrast does not isolate learned weighting from multi-candidate averaging.

On ShadowHand, *w/o Filter* had the lowest mean Last-10 return, -242.4, compared with TCGI's -112.2, and the largest SD, 121.7 (Table 3). The no-bonus and no-penalty variants had Last-10 returns of -181.5 and -154.8, respectively. TCGI therefore had the highest Last-10 mean among these configurations, whereas the no-bonus variant had the highest AUC, -204.8 versus TCGI's -208.2. The ranking depended on whether the summary covered the final ten rounds or the full recorded interval.

On Kitchen, Last-10 success ranged from 93.8% to 96.5% for the removal variants, compared with 96.8% for TCGI. These small separations provide little evidence for ranking component importance on this task.

5.4 Hyperparameter Sensitivity

We evaluate candidate count $K \in \{4, 16\}$, temperature $T \in \{0.5, 2\}$, bonus weight $\beta \in \{0.03, 0.3\}$, and penalty threshold $\tau \in \{-2, 0\}$ on ShadowHand. Each setting changes one default parameter and has three retained seeds. The four-seed CMIL and TCGI references in Table 4 come from the main-experiment batch; the sensitivity batch contains no matched default group. Comparisons within the sweep therefore concern the tested settings, rather than changes from a matched default.

The $K = 4$ setting had a higher Last-10 mean than $K = 16$ (-117.2 versus -142.9), while their AUCs were close (-202.8 and -201.1). The $T = 2$ setting had the lowest Last-10 mean, -291.1, and largest SD, 125.0, among the tested settings.

The $\tau = -2$ setting had the highest AUC, -190.9, alongside a Last-10 return of -165.3. The $\beta = 0.3$ setting had higher Last-10 and AUC means than $\beta = 0.03$. Within this batch, $K = 4$ led Last-10 and $\tau = -2$ led AUC.

Table 4: ShadowHand return: mean ± sample SD. Main-experiment CMIL and TCGI references use four seeds; sensitivity settings use three. Bold marks the best results in each column.

Setting	Last-10	Best	AUC
Main-experiment references			
CMIL	-143.6±24.8	-10.4±2.4	-222.5±43.5
TCGI (ours)	**-112.2±23.5**	**-9.4±1.3**	-208.2±31.1
Sensitivity settings			
$K = 4$	-117.2±53.7	-10.0±0.9	-202.8±42.1
$K = 16$	-142.9±31.0	-9.7±0.9	-201.1±19.1
$T = 0.5$	-209.6±47.5	**-9.4±1.4**	-219.0±7.1
$T = 2$	-291.1±125.0	-12.0±2.9	-250.7±33.6
$\beta = 0.03$	-147.8±22.7	-10.5±0.7	-220.2±6.1
$\beta = 0.3$	-136.8±47.1	-10.5±1.2	-206.5±9.4
$\tau = -2$	-165.3±79.5	-9.5±0.1	**-190.9±15.3**
$\tau = 0$	-167.8±26.0	-10.3±0.7	-196.4±11.8

6 Conclusion

We introduced TCGI, which incorporates expert transition preferences into latent imagination for visual imitation learning. Through candidate-state fusion, reward augmentation, and conditional disagreement penalties, TCGI guides both imagined trajectories and policy optimization. It preserves fixed-horizon imagination and executes the learned actor directly at deployment. Compared with CMIL, TCGI achieves higher mean Last-10 return and AUC on ShadowHand, with smaller differences on Kitchen. Evaluations on MetaWorld and DMC extend the assessment to additional manipulation and locomotion tasks. Ablations and sensitivity analyses show how performance varies with component configurations and guidance hyperparameters.

One limitation is the scope of the sensitivity analysis, which covers selected parameter settings on ShadowHand. Extending these comparisons to other tasks would provide a clearer understanding of how task characteristics affect parameter sensitivity. Future work will explore the application of TCGI to physical robots, examining how sensing noise and actuation variability affect transition-guided imagination in practical manipulation tasks.

Reproducibility Statement

The appendix specifies model architecture, losses, gradient paths, optimization settings, environment interfaces, and metric definitions. It provides per-seed summaries for the Kitchen and ShadowHand main comparisons, their component ablations, and the ShadowHand sensitivity settings. The training source archive contains task adapters and configurations; expert trajectories, trained checkpoints, and complete evaluation histories are not included in that archive. Appendix D describes resource accounting and source-material coverage.

AI Use Statement

AI assistance was used for language polishing and checking formatting consistency with the paper template. The reported performance values come from the experimental records.

REFERENCES

Leonardo Barcellona, Andrii Zadaianchuk, Davide Allegro, Samuele Papa, Stefano Ghidoni, and Efstratios Gavves. Dream to manipulate: Compositional world models empowering robot imitation learning with imagination. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/8f0d446441a938d9de420a8ab8d7fd36-Abstract-Conference.html.

Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. In *Advances in Neural Information Processing Systems*, volume 31, 2018. URL https://proceedings.neurips.cc/paper/2018/hash/3de568f8597b94bda53149c7d7f5958c-Abstract.html.

Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In *Advances in Neural Information Processing Systems*, volume 26, 2013. URL https://proceedings.neurips.cc/paper/2013/hash/af21d0c97db2e27e13572cbf59eb343d-Abstract.html.

Robert Dadashi, Léonard Hussenot, Matthieu Geist, and Olivier Pietquin. Primal Wasserstein imitation learning. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=TtYSU29zgR.

Branton DeMoss, Paul Duckworth, Jakob Foerster, Nick Hawes, and Ingmar Posner. DITTO: Offline imitation learning with world models. arXiv preprint arXiv:2302.03086v2, 2025. URL https://arxiv.org/abs/2302.03086v2.

Chelsea Finn, Sergey Levine, and Pieter Abbeel. Guided cost learning: Deep inverse optimal control via policy optimization. In *International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pp. 49–58. PMLR, 2016. URL https://proceedings.mlr.press/v48/finn16.html.

Justin Fu, Katie Luo, and Sergey Levine. Learning robust rewards with adversarial inverse reinforcement learning. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rkHywl-A-.

Divyansh Garg, Shuvam Chakraborty, Chris Cundy, Jiaming Song, and Stefano Ermon. IQ-Learn: Inverse soft-Q learning for imitation. In *Advances in Neural Information Processing Systems*, volume 34, pp. 4028–4039, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/210f760a89db30aa72ca258a3483cc7f-Abstract.html.

Seyed Kamyar Seyed Ghasemipour, Richard Zemel, and Shixiang Gu. A divergence minimization perspective on imitation learning methods. In *Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pp. 1259–1277. PMLR, 2020. URL https://proceedings.mlr.press/v100/ghasemipour20a.html.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, volume 27, 2014. URL https://proceedings.neurips.cc/paper_files/paper/2014/hash/f033ed80deb0234979a61f95710dbe25-Abstract.html.

Abhishek Gupta, Vikash Kumar, Corey Lynch, Sergey Levine, and Karol Hausman. Relay policy learning: Solving long-horizon tasks via imitation and reinforcement learning. In *Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pp. 1025–1037. PMLR, 2020. URL https://proceedings.mlr.press/v100/gupta20a.html.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2555–2565. PMLR, 2019. URL https://proceedings.mlr.press/v97/hafner19a.html.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*, 2020. URL https://iclr.cc/virtual_2020/poster_S1lOTC4tDS.html.

Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering Atari with discrete world models. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=0oabwyZbOu.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse control tasks through world models. *Nature*, 640:647–653, 2025. doi: 10.1038/s41586-025-08744-2.

Siddhant Haldar, Vaibhav Mathur, Denis Yarats, and Lerrel Pinto. Watch and match: Supercharging imitation with regularized optimal transport. In *Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pp. 32–43. PMLR, 2023. URL https://proceedings.mlr.press/v205/haldar23a.html.

Nicklas Hansen, Xiaolong Wang, and Hao Su. Temporal difference learning for model predictive control. In *International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 8387–8406. PMLR, 2022. URL https://proceedings.mlr.press/v162/hansen22a.html.

Nicklas Hansen, Hao Su, and Xiaolong Wang. TD-MPC2: Scalable, robust world models for continuous control. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/cf73d57b6dcda32b293df7c2d5341f49-Abstract-Conference.html.

Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. In *Advances in Neural Information Processing Systems*, volume 29, 2016. URL https://proceedings.neurips.cc/paper/2016/hash/cc7e2b878868cbae992d1fb743995d8f-Abstract.html.

Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. In *Advances in Neural Information Processing Systems*, volume 32, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/5faf461eff3099671ad63c6f3f094f7f-Abstract.html.

Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. MOReL: Model-based offline reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 33, pp. 21810–21823, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/f7efa4f864ae9b88d43527f4b14f750f-Abstract.html.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015. URL https://arxiv.org/abs/1412.6980.

Victor Kolev, Rafael Rafailov, Kyle Hatch, Jiajun Wu, and Chelsea Finn. Efficient imitation learning with conservative world models. In *6th Annual Learning for Dynamics & Control Conference*, volume 242 of *Proceedings of Machine Learning Research*, pp. 1777–1790. PMLR, 2024. URL https://proceedings.mlr.press/v242/kolev24a.html.

Ilya Kostrikov, Kumar Krishna Agrawal, Debidatta Dwibedi, Sergey Levine, and Jonathan Tompson. Discriminator-actor-critic: Addressing sample inefficiency and reward bias in adversarial imitation learning. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Hk4fpoA5Km.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative Q-learning for offline reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 33, pp. 1179–1191, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/0d2b2061826a5df3221116a5085a6052-Abstract.html.

Alex X. Lee, Anusha Nagabandi, Pieter Abbeel, and Sergey Levine. Stochastic latent actor-critic: Deep reinforcement learning with a latent variable model. In *Advances in Neural Information Processing Systems*, volume 33, pp. 741–752, 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/hash/08058bf500242562c0d031ff830ad094-Abstract.html.

Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *Journal of Machine Learning Research*, 17(39):1–40, 2016. URL https://jmlr.org/papers/v17/15-522.html.

Shangzhe Li, Zhiao Huang, and Hao Su. Reward-free world models for online imitation learning. In *International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 34702–34724. PMLR, 2025. URL https://proceedings.mlr.press/v267/li25af.html.

Fan-Ming Luo, Tian Xu, Xingchen Cao, and Yang Yu. Reward-consistent dynamics models are strongly generalizable for offline reinforcement learning. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/22bbfc36cb32d0baf7cd1c832da6f5e9-Abstract-Conference.html.

Anusha Nagabandi, Kurt Konolige, Sergey Levine, and Vikash Kumar. Deep dynamics models for learning dexterous manipulation. In *Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pp. 1101–1112. PMLR, 2020. URL https://proceedings.mlr.press/v100/nagabandi20a.html.

Dean A. Pomerleau. ALVINN: An autonomous land vehicle in a neural network. In *Advances in Neural Information Processing Systems*, volume 1, 1988. URL https://proceedings.neurips.cc/paper/1988/hash/812b4ba287f5ee0bc9d43bbf5bbe87fb-Abstract.html.

Rafael Rafailov, Tianhe Yu, Aravind Rajeswaran, and Chelsea Finn. Offline reinforcement learning from images with latent space models. In *Learning for Dynamics and Control*, volume 144 of *Proceedings of Machine Learning Research*, pp. 1154–1168. PMLR, 2021a. URL https://proceedings.mlr.press/v144/rafailov21a.html.

Rafael Rafailov, Tianhe Yu, Aravind Rajeswaran, and Chelsea Finn. Visual adversarial imitation learning using variational models. In *Advances in Neural Information Processing Systems*, volume 34, pp. 3016–3028, 2021b. URL https://proceedings.neurips.cc/paper/2021/hash/1796a48fa1968edd5c5d10d42c7b1813-Abstract.html.

Rafael Rafailov, Kyle Beltran Hatch, Victor Kolev, John D. Martin, Mariano Phielipp, and Chelsea Finn. MOTO: Offline pre-training to online fine-tuning for model-based robot learning. In *Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 3654–3671. PMLR, 2023. URL https://proceedings.mlr.press/v229/rafailov23a.html.

Marc Rigter, Bruno Lacerda, and Nick Hawes. RAMBO-RL: Robust adversarial model-based offline reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 35, pp. 16082–16097, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/6691c5e4a199b72dffd9c90acb63bcd6-Abstract.html.

Stephane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pp. 627–635. PMLR, 2011. URL https://proceedings.mlr.press/v15/ross11a.html.

Ramanan Sekar, Oleh Rybkin, Kostas Daniilidis, Pieter Abbeel, Danijar Hafner, and Deepak Pathak. Planning to explore via self-supervised world models. In *International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 8583–8592. PMLR, 2020. URL https://proceedings.mlr.press/v119/sekar20a.html.

Yihao Sun, Jiaji Zhang, Chengxing Jia, Haoxin Lin, Junyin Ye, and Yang Yu. Model-Bellman inconsistency for model-based offline reinforcement learning. In *International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 33177–33194. PMLR, 2023. URL https://proceedings.mlr.press/v202/sun23q.html.

Emanuel Todorov, Tom Erez, and Yuval Tassa. MuJoCo: A physics engine for model-based control. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033, 2012. doi: 10.1109/iros.2012.6386109.

Saran Tunyasuvunakool, Alistair Muldal, Yotam Doron, Siqi Liu, Steven Bohez, Josh Merel, Tom Erez, Timothy Lillicrap, Nicolas Heess, and Yuval Tassa. dm_control: Software and tasks for continuous control. *Software Impacts*, 6:100022, 2020. doi: 10.1016/j.simpa.2020.100022.

Shenghua Wan, Yucen Wang, Minghao Shao, Ruying Chen, and De-Chuan Zhan. SeMAIL: Eliminating distractors in visual imitation via separated models. In *International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 35426–35443. PMLR, 2023. URL https://proceedings.mlr.press/v202/wan23c.html.

Manuel Watter, Jost Tobias Springenberg, Joschka Boedecker, and Martin Riedmiller. Embed to control: A locally linear latent dynamics model for control from raw images. In *Advances in Neural Information Processing Systems*, volume 28, 2015. URL https://proceedings.neurips.cc/paper/2015/hash/a1afc58c6ca9540d057299ec3016d726-Abstract.html.

Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-World: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pp. 1094–1100. PMLR, 2020a. URL https://proceedings.mlr.press/v100/yu20a.html.

Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. MOPO: Model-based offline policy optimization. In *Advances in Neural Information Processing Systems*, volume 33, pp. 14129–14142, 2020b. URL https://proceedings.neurips.cc/paper/2020/hash/a322852ce0df73e204b7e67cbbef0d0a-Abstract.html.

Tianhe Yu, Aviral Kumar, Rafael Rafailov, Aravind Rajeswaran, Sergey Levine, and Chelsea Finn. COMBO: Conservative offline model-based policy optimization. In *Advances in Neural Information Processing Systems*, volume 34, pp. 28954–28967, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/f29a179746902e331572c483c45e5086-Abstract.html.

Sheng Yue, Guanbo Wang, Wei Shao, Zhaofeng Zhang, Sen Lin, Ju Ren, and Junshan Zhang. CLARE: Conservative model-based reward learning for offline inverse reinforcement learning. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=5aT4ganOd98.

Marvin Zhang, Sharad Vikram, Laura Smith, Pieter Abbeel, Matthew J. Johnson, and Sergey Levine. SOLAR: Deep structured representations for model-based reinforcement learning. In *International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 7444–7453. PMLR, 2019. URL https://proceedings.mlr.press/v97/zhang19m.html.

Wenjia Zhang, Haoran Xu, Haoyi Niu, Peng Cheng, Ming Li, Heming Zhang, Guyue Zhou, and Xianyuan Zhan. Discriminator-guided model-based offline imitation learning. In *Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pp. 1266–1276. PMLR, 2023. URL https://proceedings.mlr.press/v205/zhang23c.html.

A Additional Connections to Prior Work

A.1 What demonstrations supervise

Behavior cloning learns actions from demonstrations, from early neural driving policies (Pomerleau, 1988) to visuomotor control. Distribution shift arises when these policies act along their own trajectories (Ross et al., 2011). Guided policy search connects supervised policy fitting to end-to-end visuomotor learning (Levine et al., 2016). TCGI uses demonstrations for actor BC targets, state action discriminator positives, and transition scorer positives. The transition tuples add successor information to action supervision.

Reward learning derives control objectives from demonstrations. Guided cost learning jointly learns nonlinear costs and improves a policy (Finn et al., 2016); GAIL and AIRL use adversarial objectives (Ho & Ermon, 2016; Fu et al., 2018); IQ-Learn represents reward and policy through a learned Q-function (Garg et al., 2021). The divergence-minimization view relates these objectives through comparisons of expert and learner distributions (Ghasemipour et al., 2020). TCGI's added scorer compares complete latent transition tuples. Its density-ratio interpretation in Appendix B.5 follows the optimal-discriminator argument for adversarial generative learning (Goodfellow et al., 2014).

A.2 Transport rewards and the ROT baseline

Primal Wasserstein Imitation Learning constructs imitation objectives from transport between expert and agent state action distributions (Dadashi et al., 2021). Sinkhorn scaling computes entropically regularized transport (Cuturi, 2013). ROT combines transport-based online rewards with BC pretraining and adaptive behavioral regularization through a soft Q-filter (Haldar et al., 2023). It thus provides a comparison with both offline BC and world-model imitation. The archived ROT adapter stabilizes transport computation. Whereas ROT aligns agent and expert trajectories, TCGI applies softmax weights to successor candidates conditioned on the same current latent state and action.

A.3 Representations, imagination, and planning

Predictive latent representations support different control procedures. Embed to Control constrains latent dynamics to be locally linear (Watter et al., 2015); SOLAR learns representations for locally fitted dynamics and costs (Zhang et al., 2019). Stochastic Latent Actor-Critic learns sequential representations for model-free policy updates (Lee et al., 2020). PlaNet plans through recurrent latent dynamics, while Dreamer trains an actor through imagined trajectories (Hafner et al., 2019; 2020). DreamerV2 introduces discrete latents and KL balancing (Hafner et al., 2021). TCGI retains CMIL's continuous RSSM and the balanced KL objective in Equation (11).

V-MAIL uses latent imagination for adversarial visual imitation (Rafailov et al., 2021b). SeMAIL separates task-relevant dynamics from visual distractors (Wan et al., 2023). TCGI addresses successor construction within the inherited representation objective, a different intervention from learning distractor-invariant features.

Task-oriented latent planning offers another use of world models. TD-MPC combines short-horizon trajectory optimization with a learned terminal value (Hansen et al., 2022); TD-MPC2 develops a scalable, decoder-free approach (Hansen et al., 2024). Reward-free world models combine inverse soft-Q learning with planning for online imitation (Li et al., 2025). TCGI generates candidates during policy training and executes its actor directly at evaluation time.

A.4 How uncertainty enters model use

Probabilistic ensembles with trajectory sampling propagate uncertainty during planning (Chua et al., 2018). Model-based policy optimization uses short rollouts branched from real data to limit model bias (Janner et al., 2019), and latent offline model-based policy optimization represents uncertainty in latent space (Rafailov et al., 2021a). Plan2Explore uses latent disagreement to encourage exploration (Sekar et al., 2020). Ensemble statistics can therefore encourage exploration or penalize uncertain predictions, depending on the learning objective.

Other forms of conservatism act on values or dynamics: conservative Q-learning regularizes values (Kumar et al., 2020), while RAMBO optimizes adversarial dynamics subject to fitting observed transitions (Rigter et al., 2022). TCGI conditions CMIL's disagreement penalty on a learned transition score. The score expresses a preference relative to expert transition data; disagreement measures dispersion among model predictions. Both signals enter the training reward.

B Implementation Details

B.1 Training procedure

Algorithm 1 gives the outer training loop; Algorithm 2 expands one learner update, and Algorithm 3 specifies the shared guided-imagination routine. The input replay already contains demonstrations. For a fresh run, the shared runner first makes one learner call to materialize the networks and optimizers, then performs 1,000 learner updates, 100 BC warm-up updates (each also updates the world model), and 5,000 actor interaction steps to seed online replay. During online learning it schedules two learner updates every five interaction steps. BC warm-up uses its separate SGD optimizer (learning rate 2×10^{-4}); the full learner uses Adam. Restored runs follow their runner's recovery path.

Algorithm 2 One TCGI learner update

Require: Replay sequence batch B, expert sequence batch E, scorer-update counter n

Infer replay posteriors S^B and minimize $\mathcal{L}_{\mathrm{WM}}$ (Eq. (10)) over θ.
Retain detached S^B from this forward pass as rollout roots.
Infer expert posteriors z^E with the updated world model.
Generate H unfiltered actor transitions per root, without gradients.
Form expert pairs (z_j^E, a_{j+1}^E) and policy pairs (z_t, a_t).
Add Gaussian noise to both concatenated pair inputs; update ω (Eq. (2)).
Form expert tuples $(z_j^E, a_{j+1}^E, z_{j+1}^E)$ and unfiltered policy tuples.
Detach tuples; add independent Gaussian noise to endpoints, leaving actions unchanged.
Minimize the head-average of Eq. (3) over the active scorer parameters.
Increment n; every 500 scorer updates, copy the active ensemble into a three-slot FIFO queue.
Freeze world-model, discriminator, scorer, and critic parameters; enable actor parameters.
$\mathcal{R}_\pi \leftarrow \textsc{GuidedRollout}(S^B, H)$, retaining action and dynamics gradients.
Evaluate $Q_t = \min_i Q_{\eta_i}(z_t, a_t)$ and compute G_t^λ (Eq. (12)).
Update ψ using the actor loss and aligned expert actions (Eq. (9)).
Freeze the actor; enable critic parameters.
$\mathcal{R}_Q \leftarrow \textsc{GuidedRollout}(S^B, H)$ with the updated actor, without gradients.
Draw one target-critic index I uniformly from $\{1, 2\}$ for this update.
Compute $\bar{G}_t^\lambda$ with $\bar{Q}_{\eta_I}$ and Eq. (12); detach the targets.
Form replay targets y_j^B from imitation reward and model continuation (Eq. (14)).
Update both critics using $\mathcal{L}_Q$ (Eq. (15)).
Copy critics to targets after the first update and every 100 updates thereafter.
return Updated parameters, historical scorer queue, and counter n.

Each RSSM call samples one Gaussian prior head and then its stochastic state; the chosen head is shared across the call's batch. Scorer endpoints are detached, but action derivatives remain available. In the actor phase, fused states retain derivatives through both candidate tensors and fusion weights. Only the scalar score summary detaches the weights. The threshold comparison has no gradient. No score threshold terminates the rollout or changes its discount. The routine reduces to one unfiltered successor when $K = 1$, while still computing its score.

Algorithm 2 distinguishes replay posteriors saved before the world-model optimizer step from expert posteriors inferred afterward, matching the implementation. Incoming-action indexing uses a_{j+1}^E for the transition from z_j^E to z_{j+1}^E; imagined a_t is indexed by its departure state. The optional expert-rollout uncertainty diagnostic does not enter any loss and is omitted from the pseudocode.

Algorithm 3 GUIDEDROLLOUT: the fixed-horizon imagination operator

Require: Detached replay posterior roots S_0, horizon H, actor π_ψ, trained scoring heads

for $t = 0, \dots, H-1$ **do**
  $z_t \leftarrow \mathrm{feat}(S_t)$; sample $a_t \sim \pi_\psi(\cdot \mid \mathrm{sg}(z_t))$.
  for $k = 1, \dots, K$ **do**
    Call the RSSM transition on (S_t, a_t) to obtain $S_{t+1}^{(k)}$.
    $v_t^{(k)} \leftarrow \bar{r}_\phi(\mathrm{sg}(z_t), a_t, \mathrm{sg}(\mathrm{feat}(S_{t+1}^{(k)})))$ (Eqs. (4), (5)).
  end for
  $w_t \leftarrow \mathrm{softmax}(v_t/T)$ (Eq. (5)).
  Fuse every RSSM tensor field with w_t; form c_t with detached weights (Eq. (6)).
  Compute disagreement u_t from the fused record (Eq. (7)).
  $\widetilde{r}_t \leftarrow D_\omega(z_t, a_t) + \beta c_t - \alpha \mathbf{1}\{\mathrm{sg}(c_t) < \tau\} u_t$ (Eq. (8)).
  Store $(z_t, a_t, \widetilde{r}_t, S_{t+1})$; keep discount γ at every step.
end for
return H transition records; the final state action value bootstraps the first $H-1$ rewards.

B.2 ARCHITECTURES AND OPTIMIZATION

The continuous RSSM uses a 400-dimensional deterministic state and a 360-dimensional stochastic state, producing a 760-dimensional feature. Its prior has ten Gaussian heads. Each transition call randomly selects one prior head and samples its Gaussian; the selected head is shared across that call's batch. Repeated candidate calls can select different heads. Standard deviations are parameterized as $2\,\mathrm{sigmoid}(v/2) + 0.1$, with network output v.

The image encoder has four convolutional layers with base depth 48 and kernel sizes $(4, 4, 4, 4)$. The decoder uses kernels $(5, 5, 6, 6)$. The actor and each of two critics use four hidden layers of width 400 with exponential linear unit (ELU) activations. The continuous actor outputs a truncated normal distribution, with minimum standard deviation 0.1. Each critic is a unit-variance Gaussian regression head on the concatenated feature and action. The imitation discriminator has two hidden layers of width 200 and a Bernoulli output.

Each transition scorer has three hidden layers of width 256 and ELU activations, with input dimension $2 \times 760 + d_a$, where d_a is the task action dimension. There are three active heads and up to three stored snapshots of that ensemble. Scoring therefore averages between three and twelve logits. The active ensemble is copied to the oldest historical slot every 500 scorer updates; empty slots are excluded. Historical parameters remain frozen while their outputs retain action derivatives.

All full-learner optimizers are Adam (Kingma & Ba, 2015); the separate BC warm-up uses SGD as specified in Appendix B.1. Training uses 32-bit precision in the archived task configurations. The scalar coefficient stored as `unc_penalty` is -10; Equation (8) uses the equivalent positive coefficient $\alpha = 10$. The configuration names `dyn_softmax_temp` and `dyn_r_safe` correspond to T and τ, respectively. The scorer's ensemble size is distinct from both the world-model ensemble size M and candidate count K. Figure 1 summarizes the data flow; its weighted state sum applies to every RSSM tensor field. Gradient paths and update details are given in Algorithms 2–3.

B.3 WORLD-MODEL, ACTOR, AND CRITIC OBJECTIVES

Let $p_{\theta,m}$ denote the selected Gaussian prior head and q_θ the posterior in a replay sequence. The image decoder distribution is $p_\theta(o_t \mid z_t)$. With expectations over sampled sequences, the implemented world-model objective is

$$\mathcal{L}_{\mathrm{WM}} = -\mathbb{E} \log p_\theta(o_t \mid z_t) + \mathcal{L}_{\mathrm{KL}}, \tag{10}$$

$$\mathcal{L}_{\mathrm{KL}} = 0.2 \max\{1, \mathbb{E}\,\mathrm{KL}(q_\theta \| \,\mathrm{sg}(p_{\theta,m}))\} + 0.8 \max\{1, \mathbb{E}\,\mathrm{KL}(\mathrm{sg}(q_\theta) \| p_{\theta,m})\}. \tag{11}$$

KL balancing follows the world-model training approach of Hafner et al. (2021). The free-nats threshold is applied after averaging in the configured `free_avg` branch. Reconstruction and KL both have unit loss weight. The configured model has no environment-reward prediction head and no learned discount head. Imagination uses a constant discount γ.

Table 5: Shared TCGI configuration. Task-specific sequence lengths and evaluation schedules are described in Appendix C.

Parameter	Value
World-model / actor / critic learning rate	3×10^{-4}
Imitation discriminator / scorer learning rate	3×10^{-5}
Optimizer	Adam
Optimizer epsilon / weight decay	10^{-5} / 10^{-6}
Gradient clipping norm	100
Batch size	32 sequences
Discount γ / trace λ	0.99 / 0.95
Entropy coefficient η_{ent} / BC weight η_{BC}	10^{-4} / 10
Discriminator input-noise SD	2.5
Target-critic update interval / replacement fraction	100 updates / 1.0
Imagination horizon H / candidate count K	15 / 8
Temperature T / bonus weight β	1.0 / 0.1
Penalty threshold τ / coefficient α	-1.0 / 10
Active scorer heads / historical snapshots	3 / 3
Scorer snapshot interval	500 updates

For actor learning, $Q_t = \min_{i=1,2} Q_{\eta_i}(z_t, a_t)$ and the trace parameter is $\lambda = 0.95$. Initialize $G^\lambda_{H-1} = Q_{H-1}$ and recurse backward for $t = H-2, \dots, 0$:

$$G^\lambda_t = \widetilde{r}_t + \gamma\big[(1-\lambda)Q_{t+1} + \lambda G^\lambda_{t+1}\big]. \tag{12}$$

The actor uses $V^\lambda_t = (1-\lambda)Q_t + \lambda G^\lambda_t$ in Equation (9). The H generated transitions supply H state action values, but only the first $H-1$ rewards enter this loss; Q_{H-1} is the bootstrap. The supervised term is $\mathcal{L}_{\text{BC}} = -\mathbb{E}_{(z,a)\sim\mathcal{D}_E} \log \pi_\psi(a \mid z)$, using the incoming-action alignment in Appendix B.1. The entropy term is recomputed from the actor at detached imagined features. Actor gradients pass through sampled actions, imagined dynamics, and the score paths in Equations (5)–(6); all other network parameters remain fixed during this phase.

For each critic update, draw I uniformly from $\{1, 2\}$ and use the target critic $\bar{Q}_{\eta_I}$ for every bootstrap in that update. A fresh guided rollout supplies $\bar{Q}_t = \bar{Q}_{\eta_I}(z_t, a_t)$ and targets $\bar{G}^\lambda_t$ computed by Equation (12) with Q replaced by $\bar{Q}$. The imagined critic loss is, up to the fixed Gaussian normalization constant,

$$\mathcal{L}^{\text{imag}}_Q = \frac{1}{4}\sum_{i=1}^{2} \mathbb{E}^{H-2}_{t=0}\left[\gamma^{t+1}\big(Q_{\eta_i}(z_t, a_t) - \text{sg}(\bar{G}^\lambda_t)\big)^2\right]. \tag{13}$$

The factor $1/4$ combines averaging over two critics with the $1/2$ in unit-variance Gaussian negative log-likelihood.

The real-data term operates along the replay sequence used to seed imagination. Let j index this sequence, $(z^B_j, a^B_j, z^B_{j+1})$ be an observed transition, $\bar{Q}^{(j+1)}_0$ the target value for the first imagined action rooted at its successor, and $\bar{G}^{(j+1)}_0$ the corresponding model continuation target. For a replay sequence of length L, use $j = 0, \dots, L-3$ and interpret a^B_j as its outgoing action (stored at array index $j+1$). The target is

$$y^B_j = D_\omega(z^B_j, a^B_j) + \gamma\big[(1-\lambda)\bar{Q}^{(j+1)}_0 + \lambda\bar{G}^{(j+1)}_0\big]. \tag{14}$$

Writing $e^B_{i,j} = Q_{\eta_i}(z^B_j, a^B_j) - \text{sg}(y^B_j)$ for the replay prediction error, the total critic loss (up to constant Gaussian terms) is

$$\mathcal{L}_Q = \mathcal{L}^{\text{imag}}_Q + \frac{1}{4}\sum_{i=1}^{2} \mathbb{E}_j[(e^B_{i,j})^2]. \tag{15}$$

Continuation evaluates the imagined policy action at the observed successor, not the next recorded replay action. Replay uses the discriminator reward without the bonus or disagreement penalty, followed by guided model continuation; imagined targets use the guided reward throughout. Both regression terms have equal loss weights and share critic parameters, forming a joint surrogate objective rather than the exact value of a single stationary reward function. Target critics are hard-copied every 100 updates.

B.4 Indexing, gradient paths, and update order

The archive stores *incoming* actions: array entry $\mathrm{action}[j+1]$ maps observation j to observation $j+1$. Expert scorer tuples therefore use `feat[:, :-1]`, `action[:, 1:]`, and `feat[:, 1:]`. The expert BC objective uses the same pairing. In guided rollouts, `action[1:]` contains the newly sampled actions, while `action[0]` retains the incoming replay actions. Mixing these conventions would alter both the imitation objective and transition labels.

For a replay sequence of length L, feature indices $0, \ldots, L-3$ pair with stored action indices $1, \ldots, L-2$. Continuation starts at successor indices $1, \ldots, L-2$, giving $L-2$ critic targets. The final reward among $L-1$ adjacent pairs is dropped by the sequence slice. This truncation is independent of environment-terminal masking; continuation uses fixed γ.

A rollout stores $H+1$ states and H new actions. Actor and critic losses use the first $H-1$ rewards and the last available state action value as bootstrap; $H = 15$ therefore gives 14 reward terms. The transition-score sequence has a padded first entry, which is skipped when composing rewards.

Gradient flow follows four rules. First, actor inputs z_t are detached when sampling actions, while the recurrent imagined-state graph remains available through the dynamics. Second, scorer endpoints are detached, but the action input remains differentiable during actor learning. Third, fusion weights are differentiable in the state-fusion path; they are detached only in the scalar c_t used by the reward. Fourth, the threshold comparison uses detached c_t and has no gradient. World-model, scorer, discriminator, and critic parameters are fixed during the actor phase. A separate no-gradient rollout then supplies the critic phase. Scorer training perturbs the two latent endpoints with independent Gaussian noise of SD 2.5, whereas imitation-discriminator training perturbs its entire concatenated state action input. Neither perturbation is applied to rollout scoring.

At a fixed state action pair, the deterministic recurrence and full ensemble mean predictions are shared across candidate calls; the sampled prior head and stochastic state vary. Fusion changes the stochastic state used by later transitions while leaving the current step's ensemble-mean disagreement unchanged.

B.5 Interpretation and relation to prior formulations

The logistic classification loss in Equation (3) has a standard density-ratio interpretation (Goodfellow et al., 2014; Ho & Ermon, 2016; Ghasemipour et al., 2020). For fixed positive and negative joint training-tuple densities $p_E(\xi)$ and $p_I(\xi)$, where $\xi = (z, a, z')$ includes the configured feature perturbations, an unrestricted scalar classifier minimizes its population loss at

$$r^*(\xi) = \log \frac{p_E(\xi)}{p_I(\xi)}. \tag{16}$$

This follows by setting the derivative of $p_E \log(1 + e^{-r}) + p_I \log(1 + e^{r})$ to zero wherever both densities are positive. The statement concerns a fixed population objective. TCGI trains finite neural ensembles on changing representations and policy distributions, with noisy features and historical snapshots.

Even under the idealized population interpretation, the joint ratio includes both a state action visitation ratio and a conditional successor ratio:

$$\log \frac{p_E(z, a, z')}{p_I(z, a, z')} = \log \frac{p_E(z, a)}{p_I(z, a)} + \log \frac{p_E(z' \mid z, a)}{p_I(z' \mid z, a)}. \tag{17}$$

Expert successors are posterior features conditioned on observations; policy successors are model predictions. A high score therefore indicates compatibility with these expert training examples, not

an identified physical-error magnitude. This algebra explains the interpretation of the training signal; it supplies no TCGI convergence or return-improvement guarantee.

CMIL analyzes imitation through visitation mismatch and model discrepancy (Kolev et al., 2024). Its bounds motivate conservatism, but TCGI's gated penalty and field-wise fusion do not directly inherit them. MOREC trains an adversarial dynamics model at dataset state action pairs, averages discriminator probabilities before constructing a dynamics reward, and selects a successor from candidates (Luo et al., 2024). Its convergence analysis addresses its specified tabular optimization setting. TCGI instead averages logits of active and finite historical ensembles, uses policy-imagined negative transitions, and fuses latent records. These changes place TCGI outside the setting of MOREC's convergence analysis.

C Environments and Evaluation

C.1 Observation and demonstration contracts

The Kitchen environment is adapted from the Relay Policy Learning codebase (Gupta et al., 2020); Baoding Balls uses the PDDM task implementation (Nagabandi et al., 2020) through the inherited CMIL interface (Kolev et al., 2024). MetaWorld and DMC use their respective benchmark implementations (Yu et al., 2020a; Tunyasuvunakool et al., 2020). The model-based image interface normalizes unsigned-byte RGB pixels to $[-0.5, 0.5]$. Incoming 128×128 images are downsampled by taking every second pixel along each spatial dimension; 64×64 inputs are retained. The archived runtime specifies `dm_control==1.0.3`, `gym==0.21.0`, `mujoco_py==2.1.2.14`, and `torch==2.2.2`. MetaWorld and dexterous-manipulation code are supplied as task-specific source snapshots. Source-material coverage is summarized in Appendix D.

The Kitchen expert manifest identifies ten trajectory files. The ShadowHand visual-baseline loader specifies twenty trajectories and 2,400 action transitions; its expert frames are downsampled from 128×128 to 64×64. The DMC runner requires ten trajectories per task with 501 RGB observations and 501 incoming-action entries, the first action being zero. MetaWorld configurations reference aligned task-specific datasets, but the source-only archive does not contain their complete trajectory inventory. These contracts describe the dataset interfaces available in the source archive.

Figure 2 shows the eight evaluated tasks across the four environment families.

C.2 Interaction, evaluation, and initialization

The Kitchen and ShadowHand model-based configurations use action repeat one, ten evaluation episodes, and environment time limits of 280 and 75 steps, respectively. Their nominal evaluation intervals are 560 and 300 counter steps. Replay sequence lengths are 50 for Kitchen and 25 for ShadowHand. Available aligned Dial-turn and Sweep-into configurations use sequence length 50, time limit 100, and evaluation every 10,000 steps over ten episodes. The additional MetaWorld groups also contain historical configurations.

The DMC protocol uses action repeat two, 500 agent decisions per episode, and ten evaluation episodes every 5,000 agent steps. Its online counter records 250,000 decisions, corresponding to 500,000 simulator frames. The three tasks are Walker Walk, Walker Run, and Cheetah Run. Evaluation uses isolated random states in the dedicated runner. Success is not defined for these locomotion results. The actor's distribution mode supplies evaluation actions, with no extra action noise; latent posterior sampling remains enabled in the archived configuration.

The shared training schedule uses 1,000 full learner pretraining updates and 100 behavior-cloning warm-up updates, the latter also updating the world model. It then collects 5,000 initial policy interaction steps and performs two learner updates per five interaction steps. The shared legacy runner increments its counter during BC warm-up as well as during interaction, whereas the dedicated DMC runner tracks actual online decisions. Evaluation and step labels follow these runner definitions, so nominal budgets can differ in their interaction and optimization costs.

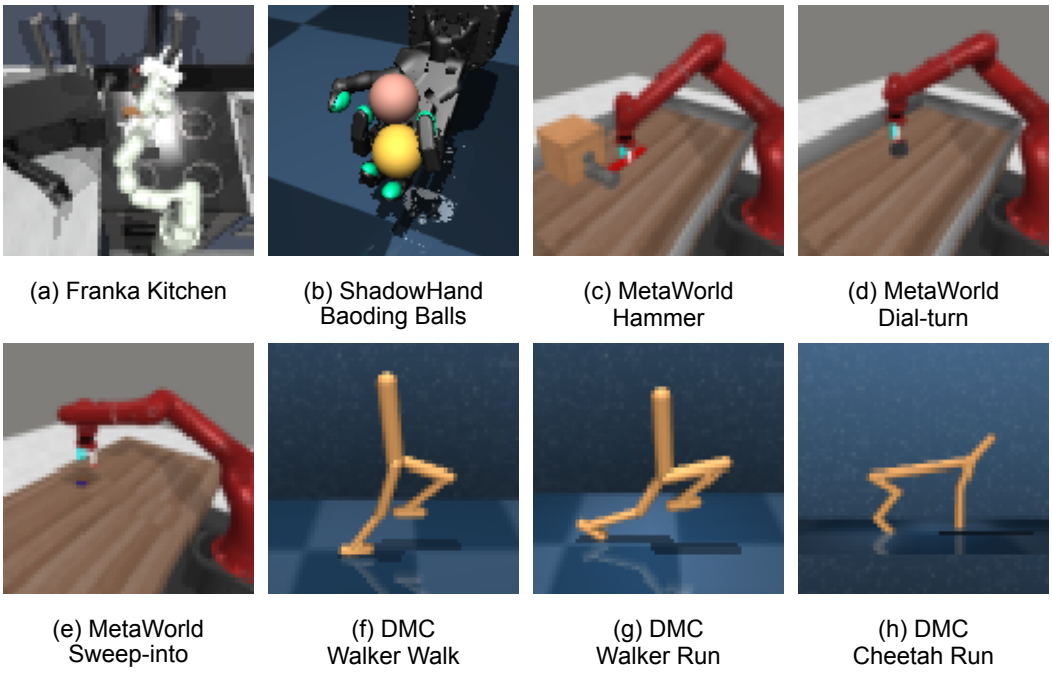


Figure 2: Example RGB observations from expert demonstrations for the eight evaluated tasks: (a) Franka Kitchen, (b) ShadowHand Baoding Balls, (c) Hammer, (d) Dial-turn, (e) Sweep-into, (f) Walker Walk, (g) Walker Run, and (h) Cheetah Run. Panels (c–e) are MetaWorld tasks; panels (f–h) are DMC tasks. Each panel shows one demonstration frame to illustrate the task scene.

C.3 Metric definitions

For one seed, let m_j be the mean task metric over evaluation episodes in round j, with J_e rounds in the record. Last-10 is the average of the last ten m_j values; Best is $\max_j m_j$; Final, reported in the full seed tables, is m_{J_e}. AUC is the recorded curve area divided by the length of the recorded training interval. This normalization preserves success or return units, and success is displayed in percent.

For per-seed metric values $v_1, \ldots, v_n$, the displayed summary is

$$\bar{v} = \frac{1}{n}\sum_i v_i, \qquad s = \sqrt{\frac{1}{n-1}\sum_i (v_i - \bar{v})^2}. \tag{18}$$

All experiments summarize four training seeds per configuration using means and sample SDs, except sensitivity configurations, which use three seeds. The CMIL and TCGI main-experiment references in the sensitivity table use four seeds. The training seed is the aggregation unit; evaluation episodes contribute to its round means. Each metric is aggregated across the training seeds. Table entries are rounded to one decimal, and returns are compared within tasks because their scales differ.

D Resource Accounting and Source Materials

Aggregate runtime. The approximately 9,300-hour estimate sums wall-clock time for training, simulation, and evaluation across runs. The accounting convention assigns three runs to each sensitivity configuration and four to other configurations, including historical variants: 38 configurations and 144 nominal runs in total. The estimate describes campaign-level computation, not elapsed calendar time, hardware-normalized GPU occupancy, or per-method speed.

Source and data scope. The source archive contains the TCGI and CMIL learners, task adapters, BC/ROT implementations, environment specifications, and ablation/sensitivity configurations. Ex-

pert trajectories, model checkpoints, and complete training histories are not included in this archive. Re-execution requires the corresponding datasets and a compatible Linux, CUDA, rendering, and MuJoCo environment.

E PER-SEED RESULTS ON FRANKA KITCHEN AND SHADOWHAND

The following tables list individual seed values for the main Franka Kitchen and ShadowHand Baoding Balls comparisons. Kitchen reports success (%); ShadowHand reports return. Mean/SD rows summarize four training seeds using Equation (18). Bold group means indicate the largest values within each task and metric; individual seed rows are not ranked.

E.1 FRANKA KITCHEN

Table 6: Franka Kitchen: per-seed success (%) and group summaries (mean ± sample SD).

Seed / group	Last-10	Best	Final	AUC
BC				
0	32.0	70.0	20.0	31.5
1	25.0	60.0	20.0	24.5
2	41.0	70.0	60.0	35.7
3	49.0	80.0	50.0	38.5
Mean/SD	36.8±10.5	70.0±8.2	37.5±20.6	32.6±6.1
CMIL				
0	97.0	100.0	90.0	74.6
1	93.0	100.0	90.0	74.5
2	97.0	100.0	100.0	76.1
3	97.0	100.0	90.0	77.4
Mean/SD	96.0±2.0	**100.0±0.0**	92.5±5.0	75.7±1.4
ROT				
0	92.0	100.0	100.0	63.2
1	44.0	100.0	50.0	26.4
2	96.0	100.0	90.0	86.4
3	98.0	100.0	100.0	48.6
Mean/SD	82.5±25.8	**100.0±0.0**	85.0±23.8	56.2±25.2
TCGI				
0	100.0	100.0	100.0	78.5
1	98.0	100.0	100.0	76.8
2	92.0	100.0	100.0	75.6
3	97.0	100.0	100.0	74.5
Mean/SD	**96.8±3.4**	**100.0±0.0**	**100.0±0.0**	**76.3±1.7**

E.2 ShadowHand Baoding Balls

Table 7: ShadowHand Baoding Balls: per-seed return and group summaries (mean ± sample SD).

Seed / group	Last-10	Best	Final	AUC
BC				
0	-416.5	-8.5	-514.9	-414.5
1	-266.1	-7.4	-514.8	-337.9
2	-467.7	-6.6	-517.1	-394.3
3	-315.8	-8.6	-513.6	-363.8
Mean/SD	-366.5±92.0	**-7.8±0.9**	-515.1±1.4	-377.6±33.7
CMIL				
0	-175.7	-12.8	-113.0	-280.3
1	-120.6	-11.6	-113.6	-187.0
2	-128.0	-9.6	-213.0	-191.0
3	-150.0	-7.4	-114.1	-231.8
Mean/SD	-143.6±24.8	-10.4±2.4	-138.4±49.7	-222.5±43.5
ROT				
0	-515.5	-14.7	-515.2	-507.9
1	-521.8	-10.9	-521.9	-421.9
2	-516.0	-12.5	-516.2	-364.4
3	-513.9	-16.1	-514.1	-500.7
Mean/SD	-516.8±3.5	-13.5±2.3	-516.9±3.5	-448.7±68.4
TCGI				
0	-135.2	-7.8	-117.7	-179.3
1	-90.4	-10.9	-116.1	-185.7
2	-129.6	-9.6	-116.3	-223.3
3	-93.5	-9.3	-112.6	-244.6
Mean/SD	**-112.2±23.5**	-9.4±1.3	**-115.7±2.2**	**-208.2±31.1**

F Additional Ablations and Sensitivity Analysis

F.1 Franka Kitchen

Table 8: Franka Kitchen: per-seed success (%) and group summaries (mean ± sample SD).

Seed / group	Last-10	Best	Final	AUC
w/o Bonus				
0	98.0	100.0	90.0	77.1
1	96.0	100.0	90.0	73.2
2	96.0	100.0	100.0	67.7
3	96.0	100.0	90.0	75.1
Mean/SD	96.5±1.0	100.0±0.0	92.5±5.0	73.3±4.1

Continued on next page

Seed / group	Last-10	Best	Final	AUC
w/o Filter				
0	98.0	100.0	100.0	73.7
1	91.0	100.0	80.0	72.8
2	97.0	100.0	90.0	74.8
3	89.0	100.0	90.0	74.4
Mean/SD	93.8±4.4	100.0±0.0	90.0±8.2	73.9±0.9
w/o Safety				
0	89.0	100.0	100.0	71.2
1	99.0	100.0	100.0	71.3
2	97.0	100.0	100.0	74.5
3	95.0	100.0	90.0	80.5
Mean/SD	95.0±4.3	100.0±0.0	97.5±5.0	74.3±4.4

F.2 SHADOWHAND BAODING BALLS

Table 9: ShadowHand Baoding Balls: per-seed return and group summaries (mean ± sample SD).

Seed / group	Last-10	Best	Final	AUC
w/o Bonus				
0	-253.9	-13.1	-168.3	-203.3
1	-127.9	-8.0	-164.1	-176.4
2	-204.9	-10.7	-167.5	-230.6
3	-139.3	-11.5	-116.9	-208.9
Mean/SD	-181.5±59.0	-10.8±2.1	-154.2±24.9	-204.8±22.3
w/o Filter				
0	-216.6	-14.2	-370.9	-207.7
1	-134.6	-10.0	-115.7	-192.5
2	-201.4	-11.9	-220.9	-197.2
3	-416.9	-11.2	-367.7	-276.1
Mean/SD	-242.4±121.7	-11.8±1.8	-268.8±123.8	-218.4±39.0
w/o Safety				
0	-115.6	-10.5	-64.0	-204.5
1	-268.9	-11.2	-210.9	-237.9
2	-88.5	-8.9	-114.0	-212.0
3	-146.1	-10.4	-117.0	-238.4
Mean/SD	-154.8±79.7	-10.3±1.0	-126.5±61.3	-223.2±17.5

F.3 SENSITIVITY PROTOCOL

Each sensitivity group contains three seeds. Table 4 presents the four-seed CMIL and TCGI main-experiment results as separate references, with individual values in Appendix E.2.

F.4 SHADOWHAND SENSITIVITY

Table 10: ShadowHand sensitivity: per-seed return and group summaries (mean ± sample SD).

Seed / group	Last-10	Best	Final	AUC
$\beta = 0.03$				
0	-173.7	-9.8	-212.4	-221.0
1	-131.2	-10.7	-114.5	-225.8
2	-138.6	-11.0	-64.9	-213.7
Mean/SD	-147.8±22.7	-10.5±0.7	-130.6±75.1	-220.2±6.1
$\beta = 0.3$				
0	-160.7	-11.2	-63.1	-199.4
1	-82.6	-9.2	-162.6	-202.8
2	-167.2	-11.2	-264.0	-217.2
Mean/SD	-136.8±47.1	-10.5±1.2	-163.2±100.5	-206.5±9.4
$\tau = -2$				
0	-200.4	-9.5	-212.1	-174.1
1	-221.1	-9.5	-171.8	-204.0
2	-74.4	-9.4	-166.5	-194.8
Mean/SD	-165.3±79.5	-9.5±0.1	-183.5±24.9	-190.9±15.3
$\tau = 0$				
0	-175.2	-10.8	-161.3	-190.3
1	-189.3	-9.5	-113.9	-210.1
2	-138.9	-10.6	-164.2	-189.0
Mean/SD	-167.8±26.0	-10.3±0.7	-146.5±28.3	-196.4±11.8
$K = 16$				
0	-149.9	-9.1	-68.6	-222.5
1	-169.8	-10.8	-162.8	-185.9
2	-109.0	-9.3	-221.4	-194.8
Mean/SD	-142.9±31.0	-9.7±0.9	-150.9±77.1	-201.1±19.1
$K = 4$				
0	-109.3	-11.1	-164.1	-179.7
1	-67.9	-9.7	-112.7	-177.4
2	-174.4	-9.3	-214.9	-251.4
Mean/SD	-117.2±53.7	-10.0±0.9	-163.9±51.1	-202.8±42.1
$T = 0.5$				
0	-259.4	-10.8	-214.3	-227.1
1	-164.7	-9.3	-216.5	-216.5
2	-204.6	-8.0	-163.6	-213.5
Mean/SD	-209.6±47.5	-9.4±1.4	-198.1±29.9	-219.0±7.1
$T = 2$				
0	-153.8	-10.6	-263.3	-212.5
1	-398.2	-15.4	-316.6	-275.8
2	-321.3	-10.0	-268.1	-263.7
Mean/SD	-291.1±125.0	-12.0±2.9	-282.7±29.5	-250.7±33.6