From ChatGPT

General Workflow: Proposal VLM outputs a **plan sequence** (multi-step). Dynamics model imagines execution step by step, chaining latent states. Reflection happens on the **whole imagined trajectory**.

Alternative: Reflection can also be applied *per-step*, i.e., after each imagined subtask, the reflector decides whether to keep, edit, or discard that action.

The extra symbolic outputs like `ON(cup, table)` or `IN(tea bag, cup)` are very useful. They provide **checkpoints** that (a) guide the dynamics model imagination, (b) simplify reflection, and (c) enable verification without re-running expensive rollouts.

Think of them as *relational anchors*:

- If the plan predicts `ON(cup, table)` but the imagined latent state doesn't match, that's a red flag → trigger reflection.

- They also help credit assignment: you can know *which subgoal failed* (e.g., `IN(tea bag, cup)` not achieved), instead of just seeing final task failure.

A critic is a learned function C(z_t + 1, a_t) that scores the likelihood that executing action a_t from latent state z_t will achieve its local subgoal and not block future goals. It takes in the imagined future latent state, proposed action, optional predicate/subgoal, and outputs probability of success or expected reward.

- At **each subtask of the plan**, you run the dynamics model forward once.
- The critic evaluates that transition:
  - ☐ Did the latent future state align with the predicate `ON/IN/...`?
  - ☐ Is the action executable from the current latent state?
  - ☐ Does it keep the overall plan feasible?

Example:
Plan:

1. pickplace(cup, table) → target predicate: `ON(cup, table)`
2. pickplace(tea bag, cup) → target predicate: `IN(tea bag, cup)`
3. pickplace(spoon, table) → target predicate: `ON(spoon, table)`

- After step 1 imagination, critic checks: "Is cup on table in $\hat{z}_1$?"
- After step 2, critic checks: "Is tea bag inside cup in $\hat{z}_2$?"

- After step 3, critic checks: “Is spoon on table in $\hat{z}_3$?”

If step 2 fails, reflection can adjust step 2 or re-plan from there.

So basically, if *all* steps are high-confidence → accept plan, no reflection. If any step is low-confidence → local reflection can be triggered only at that step.

Reflection should condition on:
{initial state, plan sequence, imagined states, goal, predicate satisfaction vector}

https://chatgpt.com/g/g-p-6833f60115d08191a279433ff06e36da/c/68dad0b5-1500-832b-a270-776d51d6abc1

From Claude

A multi-headed critic would output:

1. **Feasibility head**: P(subtask is physically executable | current latent state) - e.g., can the robot reach the object, is grasp pose valid?
2. **Predicate satisfaction head**: P(target predicate achieved | executing action) - e.g., will ON(cup, table) actually hold after pickplace?
3. **Non-interference head**: P(action doesn't block future subtasks | remaining plan) - e.g., does placing cup here prevent accessing tea bag later?

Each head addresses a different failure mode. Training requires different negative examples for each head.

Core Thesis:

Uncertainty-aware reflection enables robust long-horizon planning under model error. We show that explicitly modeling where the dynamics model is unreliable and triggering targeted replanning at those points achieves X% higher success on Y+ step tasks compared to filtering-based approaches.

Why strong: Addresses a real problem (compounding dynamics error) with a principled solution

What you need: Uncertainty quantification in the critic, analysis of *when* reflection helps vs. hurts, theoretical or empirical characterization of the reliability horizon

---

Focus on uncertainty and selective replanning (my recommendation)

- Don't always reflect—only when the critic is uncertain or predicts failure
- Characterize the "reliability horizon" of the dynamics model
- Show that early replanning at uncertain steps prevents catastrophic failures later

- Thesis: "Predictive uncertainty enables proactive replanning before failures occur"
- Need: uncertainty quantification, analysis of when/why replanning helps

---

https://claude.ai/chat/4c5fd6b5-02fb-46b3-96ab-4790ec59b987

My Musing

- Quantify uncertainty in the dynamic model and proactively trigger reflection—uncertainty-aware imagination.
- After each subplan imagination, trigger the critic to check for any discrepancies with the goal.
- If there's any, replan from that subplan.
- Still keep the final critiquing, so it happens at the end as well.

- Additionally, during rollout after each action/subtask, the dynamics model predicts the next latent state.
- Then compute the discrepancy between the predicted latent state and the observed latent state.
- If there's any deviation, trigger reflection/replanning.
- Repair from the current subplan.

- Could use the following to determine deviations:
    1. Euclidean in the embedding space
    2. Predicate disagreement
    3. KL divergence in distributional encoders.

- How does training happen?
- Collect a large number of positive and negative plans (and dynamic predictions).
- Reflection/Refine on negative solutions to help models (VLM & Critic) learn.
- Do we need to finetune the VLM? I doubt it.

- How do we collect this data for the Critic training?:
- Collecting positive samples is straightforward i.e, save the successful attempts.
- For negative samples, modify the plan from the VLM to be slightly incorrect, so the dynamics model produces an output that fails to align (using the Critic) with the goal symbolic predicates - [REFINER]
- Lrefine = -Sum(logM(y+|y-, goal, current obs, plan))
- Where y+ = the ground truth plan (subtasks and goal symbolic predicates)
  y- = the negative/incorrect plan (only subtasks)

- Negative samples could also be generated by asking the VLM to generate incorrect plans (subtasks) - perhaps based on a rule, while ensuring the goal symbolic predicates are correct.

- The critic model would take in:
    1. Output from the dynamics model - I don't know yet what this would be exactly i.e either the latent state (zt+1) or the transformed observation (ot+1).
    2. The goal symbolic predicates
    3. The plan (subtasks) - both correct and incorrect??

- The loss fn for the critic model could be:
  Lcritic = -logP(f(u)|x, u), where uEz, z' - [REFINER]

REFINER: https://arxiv.org/pdf/2304.01904