30-Day Plan for Submission-Ready Bipedal Curriculum Learning Codebase

Smallest-Win Ideas for Minimal-Scope Improvements

- **Idea 1: Curriculum vs. No-Curriculum Training.** *Claim:* A policy trained from scratch on Stage-2 (tilled soil) without the gradual curriculum will perform significantly worse (more falls, instability) than our 3-stage curriculum policy. *Evidence Needed:* Training a **no-curriculum baseline** on the tilled-soil environment and comparing its fall rate and tracking error to the curriculum-trained policy (expect large gap, consistent with prior work showing skipping curriculum stages degrades learning[1]). *Effort:* ~3 person-days (set up training run and monitor until convergence). *Risk:* Low – the outcome (poorer performance without curriculum) is likely, but if the baseline unexpectedly performs well, it could weaken our curriculum's claim (mitigated by emphasizing sample efficiency or stability differences).

- **Idea 2: Flat-Ground Only Baseline.** *Claim:* A policy trained only on flat ground (Stage-1) fails to traverse tilled soil, underscoring the need for exposure to complex terrain. *Evidence Needed:* Evaluate a **plane-only policy** (e.g. final Stage-1 model) on the tilled-soil environment and record falls/100m and slip rate. Prior studies show flat-floor policies cannot handle even small unevenness[2], so we expect immediate falls or high error. *Effort:* ~1 person-day (leverage existing Stage-1 model logs; run 100+ episodes on soil). *Risk:* Low – likely dramatic failure (high impact), but must ensure fair comparison (maybe allow it a similar training duration on flat as others had in total).

- **Idea 3: Smaller Policy Network Baseline.** *Claim:* Using a significantly smaller neural network for the policy will reduce performance on tilled soil, highlighting task complexity and possibly that our curriculum helps lesser-capacity models. *Evidence Needed:* Train a **reduced-capacity policy** (e.g. half the hidden layer size) under the same 3-stage training regimen and measure final performance vs. the original policy. Expect noticeable drop in stability or tracking. *Effort:* ~2 person-days (modify model config, run training). *Risk:* Low – straightforward to implement; risk is results might be only slightly worse (if task is easy enough), but even then we can report any performance gap or training instability.

- **Idea 4: Environment Randomization Ablation.** *Claim:* Turning off terrain randomization in the tilled soil (e.g. using one fixed soil pattern or uniform properties) will lead to a less robust policy that overfits to specifics, evidenced by poorer generalization and possibly odd gaits on new instances. *Evidence Needed:* Train or fine-tune a policy with **soil randomization disabled** and compare its performance variance to the fully randomized policy (e.g. test both on a novel soil instance). We anticipate the non-randomized policy struggles on any variation, echoing observations that lack of ground "softness" randomization caused brittle performance[3]. *Effort:* ~1 person-day (just a config switch like `--soil_rand=off`, then a short fine-tune or eval). *Risk:* Low – mainly requires careful evaluation; if difference is small, we can note that randomization still provides safety margin or simplifies real deployment.

- **Idea 5: Add Stage-3 Disturbance Training.** *Claim:* Introduce a final curriculum stage with random push perturbations applied to the robot during training to improve disturbance recovery (since we can't add plants, we add impulses). This **Stage-3 push curriculum** should yield a policy that can withstand shoves or irregular forces with fewer falls, improving robustness. *Evidence*

Needed: Fine-tune the Stage-2 policy for a short Stage-3 where random forces (e.g. ±50 N impulses at the torso) occur, then test with and without perturbation training by applying pushes in evaluation to see which policy stays up. Prior work used such perturbation curriculum to improve robustness[4], so we expect a measurable drop in falls when disturbances occur. *Effort:* ~3 person-days (implement force injection in IsaacSim, run ~N=1-2e6 steps of additional training). *Risk:* Moderate – slight code integration needed (IsaacSim API to apply forces); potential tuning of push magnitude and training length. If not tuned, policy might become overly cautious or still fail under extreme push, but even partial success is a win and can be qualitatively shown (video of recovery).

- **Idea 6: Shorter Curriculum Ablation.** *Claim:* Omitting one stage of the curriculum (either the initial easy stage or the final robustness stage) will degrade performance, reaffirming the necessity of the full 3-stage progression. *Evidence Needed:* Compare full 3-stage training vs. a **2-stage variant** (e.g. Stage-1 → Stage-2 only, skipping Stage-3 perturbations). We expect metrics to worsen without the final stage (e.g. more falls under disturbances or slightly higher tracking error), aligning with known results that each curriculum stage adds value[1]. *Effort:* ~1–2 person-days (if Stage-3 is implemented as above, simply use the pre-Stage3 policy as the ablation result). *Risk:* Low – essentially uses existing models; risk is small differences that may require many trials to quantify, but even a trend can be reported.

- **Idea 7: PPO Entropy Coefficient Variation.** *Claim:* Adjusting the exploration entropy bonus in PPO (e.g. higher entropy or a schedule) can influence learning diversity and gait smoothness; we hypothesize a moderately higher entropy early on could lead to more varied footfall patterns and possibly better avoidance of local optima (or conversely, too high might slow convergence). *Evidence Needed:* Conduct a short ablation where **entropy coefficient** (e.g. `--ppo.entropy_coef=0.02` vs `0.0X` default) or schedule is changed, and compare training curves and final slip rates or tracking error. Look for differences in convergence stability or final performance. *Effort:* ~2 person-days (one extra training run and analysis of logs). *Risk:* Low – pure hyperparameter tweak; risk that results are inconclusive (no clear improvement), in which case we simply report no significant effect (still useful info for others tuning RL hyperparams for locomotion).

- **Idea 8: Foot Slip Penalty in Reward.** *Claim:* Adding a small penalty for foot slip (lateral foot velocity during stance) in the reward will encourage the policy to minimize slip, resulting in more stable footholds and lower slip rate without sacrificing speed. *Evidence Needed:* Incorporate a **slip penalty term** (e.g. -0.1 * foot_slip_velocity) in Stage-2 training and see if the slip rate metric improves (fewer slips per 100m) compared to the original reward. Check that velocity tracking and falls aren't adversely affected. *Effort:* ~3 person-days (implement sensor calc for foot slip in sim, tune weight, retrain Stage-2 or fine-tune a bit). *Risk:* Moderate – balancing the new reward term may require some tuning; too high penalty could slow movement or cause unnatural gaits, and too low might show negligible change. We must monitor that overall returns still improve or at least that slip metric changes in desired direction.

- **Idea 9: Vary Terrain Parameters for Generalization Test.** *Claim:* Demonstrating robustness to untrained terrain conditions (within Stage-2 domain) by varying physical parameters like friction or soil stiffness in evaluation will strengthen the claim of generality. *Evidence Needed:* Evaluate the trained policy on **modified soil** (e.g. 20% lower friction or slightly deeper tilled grooves than seen in training) and record performance metrics. If the policy retains similar performance, it proves generalization; if it fails, it highlights a limitation (which could be mitigated by domain randomization in training). *Effort:* ~1 person-day (adjust a few environment parameters, e.g. via

Hydra config overrides, and run eval episodes). *Risk:* Low – mostly evaluation; risk that performance drops significantly, but even that result is publishable (we can then argue for more domain randomization as future work or mention it in limitations).

- **Idea 10: Codebase Access via Container or VM.** *Claim:* Simplifying the review and reproduction process by packaging the private codebase into a self-contained Docker container or cloud VM (with IsaacSim and all dependencies pre-installed) will make our submission artifact-friendly without open-sourcing prematurely. *Evidence Needed:* A working **containerized environment** tested on a fresh machine (ensuring all training scripts and tasks run with one command). While not a research result, this is a *"small win"* for submission: it reduces friction for reviewers to validate our results. *Effort:* ~2 person-days (write a Dockerfile or setup script, ensure licensing of IsaacSim allowed in container, upload image to an anonymized registry). *Risk:* Moderate – environment setup can have surprises (NVIDIA driver, simulator licensing) so we need to test thoroughly; mitigation is to also offer a fallback like a code bundle with instructions if container fails.

(The above ideas focus on changes or tests within the existing Stage-2 soil simulation, avoiding new simulators or plant models. Each idea is scoped to be feasible within days and aims to either strengthen the paper's evidence or improve the codebase's readiness.)

Impact–Effort Matrix and Prioritization

Below we qualitatively assess each idea's Impact (benefit to the submission) vs. Effort, and we rank the top priorities:

Idea	**Impact** (H/M/L)	**Effort** (person-days)	**Priority Rank**
1. No-Curriculum baseline	High – validates necessity of curriculum (clear performance gap)	~3	**#1** (Critical)
2. Flat-ground baseline (plane-only)	High – highlights need for complex terrain exposure	~1	**#2** (Critical)
5. Stage-3 disturbance (push) training	High – adds robustness contribution and full 3-stage story	~3	**#3** (High)
4. Env randomization ablation	Medium – supports robustness claim (shows effect of randomization)	~1	#4
6. Shorter curriculum ablation	Medium – reinforces multi-stage importance (some redundancy with Idea 1)	~1	#5
8. Foot slip penalty reward tweak	Medium – improves gait quality (nice-to-have for discussion)	~3	#6
3. Smaller network baseline	Low/Medium – shows complexity, but not central to contributions	~2	#7
7. PPO entropy coefficient variation	Low – hyperparam insight, minor impact on main narrative	~2	#8
9. Vary terrain parameter eval	Low – additional generalization evidence; could be in appendix	~1	#9
10. Containerized code artifact	Medium – important for reproducibility, but not a research result	~2	#10 (parallel task outside paper content)

Justification of Top 3: We prioritize Ideas **#1, #2, and #5** as they offer the greatest "bang for buck":

- **#1 No-Curriculum Baseline:** This is essential to *defend our core claim* – if a policy cannot learn to walk on tilled soil from scratch, it justifies the curriculum strategy. Impact is huge (it directly answers a likely reviewer question, "what if you train on final task directly?") and effort is moderate but manageable. The risk is low and even a failed learning (convergence failure) is a powerful result to report.

- **#2 Flat-Only Baseline:** Demonstrating that a flat-terrain-trained policy face-plants in soil is a **clear visual and quantitative evidence** of needing tougher training environments. It reinforces that our contribution isn't trivial or solved by existing methods. Effort is minimal since we already have a flat training stage; we mostly need to evaluate it in the soil setting. This comparison will likely yield dramatic differences (e.g., >90% failure in soil[2]) that strengthen the paper's narrative.

- **#5 Stage-3 Disturbance Training:** Adding this component elevates our work from "just training on one terrain" to **"training for robustness against unexpected perturbations,"** which is valued in venues like CoRL/ICRA. It's a relatively small extension (we can reuse our training pipeline and just apply forces in simulation), yet it gives us a defensible third contribution: resilience to pushes (analogous to handling bumping into plants or uneven resistance). The impact is high because it addresses the lack of a plant-interaction stage – effectively, we replace it with random forces to simulate external disturbances. This idea does require a few extra days of implementation and training, but it's within scope and significantly strengthens the submission by rounding out the promised "3-stage curriculum."

After these, the next priorities (#4 and #6) are quick ablation studies that provide supporting evidence. Ideas #8 and #7 are more exploratory/optional – nice if time permits, but not mission-critical for submission. Idea #10 (containerized code) is handled in parallel to paper writing by allocating some time specifically for artifact prep, ensuring it doesn't detract from core experiments.

Contribution Statement (Draft, 2–3 sentences)

We present a curriculum-based deep reinforcement learning approach that enables a humanoid biped ("Hunter") to walk robustly on compliant, uneven terrain (tilled soil) using only proprioceptive inputs in simulation. Our key contribution is demonstrating that a 3-stage curriculum – progressively increasing terrain difficulty and disturbances – yields significantly improved stability and performance (fewer falls, lower tracking error) compared to training on the target terrain directly or with simpler methods. Additionally, we provide extensive analysis of design choices (e.g. environment randomization, policy capacity) and a fully reproducible codebase, establishing a clear and defensible benchmark for bipedal locomotion on challenging deformable terrain.

(This statement can be tailored to the target venue – for instance, for CoRL/ICRA we emphasize robust locomotion on novel terrain and practical training insights, whereas for NeurIPS we might highlight the curriculum learning methodology and thorough evaluation.)

Four-Week Execution Plan

- **Week 1 (Baseline Experiments & Setup):**
- *Key Tasks:* Finalize the experiment plan and baseline definitions. Implement any needed instrumentation in the code (e.g., logging falls, slip distances, contact forces) to compute the chosen metrics. Kick off baseline training runs: **(a)** no-curriculum training on Stage-2 soil (`python train.py task=tilled_soil curriculum=off seed=...`), and **(b)** plane-only training (if a fully-trained Stage-1 model is not already available or needs extending). Also prepare evaluation scripts for easy reuse (e.g., a `evaluate.py` that can run a saved policy over 100 episodes and output metrics).
- *Checkpoint (mid-week):* Both baseline training runs are launched and progressing; logging output shows that metrics (falls, error, etc.) are being recorded correctly for each episode.

- *Acceptance Criteria (end of Week 1):* **No-curriculum baseline training completes** (or at least reaches a performance plateau) and initial results confirm it underperforms the curriculum-trained policy (e.g., falls/100m notably higher). The plane-only policy is available (from existing logs or newly trained) for evaluation. We have verified that our code instrumentation works (e.g., we can see sample metrics for a test rollout). The team has all environments running deterministically (set seeds) for comparability.

- **Week 2 (Core Implementation & Experimentation):**

- *Key Tasks:* Implement the **Stage-3 perturbation curriculum** in code. This involves adding a feature to apply random pushes to the robot – for example, using IsaacSim's physics API to apply forces at random intervals (we'll add a config flag like `env.perturb.enable=true` with parameters such as force magnitude range). Start training the Stage-3 policy: initialize from the end of Stage-2 and train with perturbations for some number of iterations (monitor if convergence or performance stabilizes). In parallel, run the quick ablation experiments: **(a)** Train a policy with environment randomization off (`env.randomize=False`) – this can be a shorter training since it's more of a comparison of final performance; **(b)** (If time permits) run the smaller network training (Idea 3) or adjust an existing training to smaller network to see if it degrades; **(c)** Try a run with higher entropy (could be combined with one of the above runs for efficiency). Continue monitoring all runs.
- *Checkpoint (mid-week):* The Stage-3 perturbation training is underway and not diverging (e.g., reward is not collapsing when pushes are applied). The no-randomization variant has finished training or reached comparable training time as main runs. We have at least preliminary data from one of the ablations (say, environment random off) for analysis.
- *Acceptance Criteria (end of Week 2):* **Stage-3 perturbation policy is trained** (or at least 80% through training if it needs more time) – we should see qualitatively in simulation that the robot can sustain moderate pushes without falling. All planned ablation runs (no randomization, shorter curriculum if applicable, smaller net, entropy tweak) are completed or nearly completed, with logs saved. No critical software bugs are outstanding. At this point, we have a full set of trained models: (curriculum final policy, no-curr baseline, flat-only baseline, no-randomization policy, etc.) ready for evaluation. Any tweaks to evaluation code (e.g., computing contact force peaks) are finished and tested on a sample run.

- **Week 3 (Evaluation & Analysis):**

- *Key Tasks:* Rigorously evaluate each trained model on the standardized tests. For each policy (curriculum final, no-curr, flat-only, etc.), run multiple evaluation episodes (e.g. 100 episodes, 100m each or fixed time) to gather metrics: falls per 100m, average tracking error to commanded velocity, slip rate, peak foot contact force, etc. Use at least 3 random seeds for environment variations to get statistically meaningful results (if not already inherent in the episodes). Collect results into tables and generate plots: for example, a bar chart of falls/100m for each method, a line plot of reward curves over training for main vs baseline, etc. Perform ablation analysis: compare Stage-3 vs no-Stage-3, randomization vs none, etc., and record the differences. Begin drafting **Result sections**: interpret whether each hypothesis was confirmed. For instance, if no-curriculum failed to converge, note that; if perturbation training improved disturbance recovery by X%, document that. Also, gather any qualitative observations (save a few simulation videos or images if possible for the paper/appendix).
- *Checkpoint (mid-week):* All evaluations have been run at least once; initial metrics are computed and we have rough tables/plots. We might identify if any additional runs are needed (e.g., if one

seed was an outlier, or if we forgot a baseline). If something is missing (say we realize we should measure energy consumption or something), there's still time to run those. Team begins interpreting results and formulating the narrative (e.g., "with curriculum: 2 falls/100m vs without: 20 falls/100m – a 10x improvement").

- *Acceptance Criteria (end of Week 3):* **Complete evaluation dataset is ready** – all metrics for all experiments are finalized and triple-checked for correctness. We have produced all the figures and tables that will go into the paper (perhaps in draft form). The results support our claims (or if any didn't, we have a plan to explain them or mark as future work). By end of week, a skeleton of the paper is filled in with these results and the accompanying analysis. We should also have a clear outline of the paper's story now, backed by data.

- **Week 4 (Write-Up, Revision & Submission Prep):**

- *Key Tasks:* Focus on writing and polishing the paper and preparing the reproducibility artifacts. Write the introduction and related work sections (framing our contribution relative to e.g. prior curriculum learning in locomotion[1] and robust terrain walking results[5]). Write the methodology section clearly describing the 3-stage curriculum and any novel aspects (like the perturbation stage or reward terms). Incorporate the results from Week 3 into the paper's narrative, creating final versions of tables/figures with proper captions. Simultaneously, assemble the **reproducibility package**: clean up the code, ensure all config files (e.g., `tasks/tilled_soil.json`, hyperparameter .yaml files) are up-to-date with the used settings, and create a README with instructions. Generate exact command lines used for each experiment (e.g., for appendix: `python train.py task=tilled_soil curriculum=off seed=123` for no-curr, etc.). Prepare an anonymized submission archive or Docker image as planned (Idea 10), including model checkpoints if allowed. Finally, address **paper completeness**: write the limitations section (e.g., "Our simulation does not include plant obstacles, which we leave for future work"), ethics/broader impact notes, and ensure references are properly cited.
- *Checkpoint (mid-week):* A full draft of the paper is ready and circulated internally for feedback – including all sections from abstract to conclusion, plus placeholder references to our figures and tables. The reproducibility artifact (code package) has been tested by at least one person other than the main developer (to ensure the instructions are clear and the environment is replicable). Any missing pieces (like an appendix log snippet or a particular citation) are identified and being added.
- *Acceptance Criteria (end of Week 4):* **Submission-ready paper and artifact.** The paper is polished (proofread for clarity and formatted per target venue guidelines – e.g., PDF in IEEE or LaTeX template as required). All required items on the submission checklist (see below) are completed: figures are legible, tables have units, the video (if applicable) is prepared. The reproducibility package passes a final test: e.g., one can follow the README to train or evaluate a policy and get expected results. At this point, we are ready to submit to the conference (with a few days buffer before the actual deadline as contingency).

Evaluation Protocol

- **Metrics:** We will evaluate locomotion performance using **quantitative metrics** that reflect both success and safety: (1) *Falls per 100m traveled* – how many times the agent falls over (or episodes end due to falls) per 100 meters of forward walking, measuring stability; (2) *Tracking error* – the difference between commanded walking speed (or trajectory) and actual achieved velocity, indicating control accuracy; (3) *Slip rate* – frequency or percentage of footsteps that exhibit significant slipping (foot sliding on contact, which we derive from foot velocity data); (4) *Peak* *contact forces* – the maximum ground reaction force on each foot during walking, which we monitor to ensure the gait is smooth (lower peaks suggest less impactful, more compliant interaction with ground). All metrics will be averaged over multiple episodes to ensure statistical reliability (with standard deviations or 95% confidence where appropriate).

- **Baselines:** We compare our full curriculum-trained policy against three key baselines: **(a) No Curriculum** – a policy trained directly on the target tilled soil terrain from scratch (to test if curriculum learning is actually necessary); **(b) Plane-Only Training** – a policy trained only on flat, rigid ground and then deployed on the tilled soil (to show that without terrain-specific training, the policy fails on soft uneven soil, as noted in prior work[6]); and **(c) Smaller Network** – a policy with a reduced-capacity neural network (e.g., fewer layers or neurons) trained with the same curriculum, to see if a simpler policy can or cannot capture the required behavior. These baselines address questions of *learning method vs. environment vs. model complexity*. Each baseline will be trained (or already exists from earlier curriculum stages) under identical reward settings for a fair comparison. Performance will be reported in the same metrics as above, often in a table for side-by-side comparison with our approach.

- **Ablations:** To isolate the contributions of various design decisions, we conduct targeted ablation studies: **(1)** *Environment Randomization Off:* we train a version of the policy without any domain randomization in the soil (no variation in friction, soil geometry, etc.) to quantify how much randomization contributed to robustness. **(2)** *Shorter Curriculum:* we test the impact of removing one stage of training – for example, skipping the final disturbance training (Stage-3) to see if the robustness benefits drop, or conversely, skipping the initial easy stage to simulate training from scratch (which is essentially the no-curr baseline in (a)). This validates that each curriculum stage has a purpose[1]. **(3)** *Entropy Coefficient Change:* we run an experiment with a modified PPO entropy coefficient (e.g., increased) to see how exploration vs. convergence trade-off affects the final gait; this helps understand our hyperparameter choices. **(4)** *Reward Modification:* if we add a slip penalty or other reward term (Idea 8), we will ablate that by comparing with/without the term. Each ablation is evaluated on the same metrics to see the specific effect (e.g., "with no randomization, slip rate doubled, indicating overfitting to a specific soil condition"). These ablations will be presented likely in an Appendix table or described in text, to not distract from main results but still provide insight.

- **Experimental Setup and Seeds:** All training and evaluation will be done in **NVIDIA IsaacSim** (targeting version 2022.x or the latest stable release we started with) to simulate the locomotion environment. We ensure the physics timestep and solver settings (e.g., Δt, substeps, solver type) are consistent across experiments and we will report these details in the paper for reproducibility. To account for stochasticity in training, each policy result (for both our method and baselines) will be obtained with **≥3 random seeds** for training initialization (and environment seeds, if applicable). We will report mean and variance over these seeds for key performance metrics (e.g., reward curves, final falls/100m). For evaluation, we also vary the environment random seed for each trial so that performance isn't measured on one fixed terrain instance.

- **Runtime and Compute Budget:** Training a single policy (from scratch through the 3-stage curriculum) is expected to require on the order of **hundreds of millions of simulation steps** (for reference, comparable works used 200–380M steps for bipedal locomotion[2]). In wall-clock terms, on our hardware (e.g., an NVIDIA RTX 3090 GPU with physics acceleration), this translates to roughly *X hours per training run* (we will log the exact training time; suppose ~24-48 hours for the full curriculum run). Baselines and ablations are shorter or fewer stages, so they may train faster (e.g., no-curr might take similar steps but sometimes fails earlier; a fine-tune like Stage-3 push might be just a few extra hours). We plan our 30-day schedule to comfortably accommodate all runs in parallel or sequence given our compute resources (for instance, if we have 2 GPUs available, we can run multiple experiments concurrently). We will document the *compute used* (number of GPUs, hours, etc.) in the paper's experimental setup. All evaluations (100-episode tests) are relatively quick (minutes to an hour each) compared to training. The overall runtime budget is thus feasible within our timeframe, and we will note in the paper that experiments were conducted within this budget to reassure reviewers of practicality.

Reproducibility Package Checklist

To ensure our results can be reproduced by reviewers and the community, we will prepare a comprehensive artifact with the following components:

- **Deterministic Seeds and Settings:** We fix and document all random seeds used (for both training and evaluation). The code will allow setting these via config (e.g., seed=123) and we'll list the seeds for each experiment in a table. Physics determinism settings in IsaacSim (such as using deterministic mode for PhysX if available) will be enabled as much as possible[7]. We will also specify the exact simulator version and OS so that others can match the environment.

- **Complete Configuration Files:** All hyperparameters and environment configurations will be included. This means providing the exact **Hydra/Isaac config files** or command-line configurations for each stage. For example, if we have a `tasks/hunter_tilled_soil.json` or YAML config for the environment, it will be included, as well as any curriculum schedule description (e.g., number of iterations in Stage-1, Stage-2, Stage-3, difficulty parameters ramped, etc.). Any reward function details and neural network architecture parameters will be clearly specified (in config or in the paper appendix).

- **Explicit Training and Evaluation Commands:** We will list the exact CLI commands needed to reproduce each result. For instance, for training:

```
# Command to train the full 3-stage curriculum policy
python train.py task=tilled_soil_curriculum.yaml train_steps=300e6 headless=True
seed=42

# Command for no-curriculum baseline
python train.py task=tilled_soil_only.yaml train_steps=300e6 headless=True seed=42
curriculum=off

# Command to evaluate a trained policy on 100 episodes
python evaluate.py checkpoint=checkpoints/tilled_curriculum.pth num_episodes=100
headless=True seed=7
```

  (Note: These are illustrative; the actual commands with Hydra might have specific syntax.) Every figure or result in the paper will be traceable to a particular command or script. We will provide a mapping of experiment name → command-line in a README or appendix.

- **Training Logs and Learning Curves:** The package will include raw training logs (e.g., TensorBoard files or CSV logs of reward, etc.) for all runs, so reviewers can inspect learning progress or replicate plots. Additionally, we'll save the training **checkpoints/models** at final (and possibly intermediate stages) for direct evaluation. These might be provided in a "model zoo" directory or via a download link if files are large, with instructions.

- **Reproducibility Scripts:** We will add convenience scripts or Jupyter notebooks to reproduce our plots from the logs (to ensure no manual error in plotting). E.g., a script to plot reward vs. steps for all methods, or to compute metrics from evaluation data and generate the table that appears in the paper.

- **Environment and Dependencies:** The artifact will include an `environment.yml` or `requirements.txt` listing all Python packages and versions. We will note the required NVIDIA driver and IsaacSim version. Ideally, a Docker container will be provided (with IsaacSim and our code pre-installed, see Idea 10) so that one can run `docker run -it our_image:tag` and immediately execute the training commands in a contained environment. This container will be configured for headless mode (since reviewers may not have rendering) and include a dummy display if needed for IsaacSim.

- **Documentation (README):** A clear README.md will guide the user through setting up the environment (or using the container) and running training/evaluation. It will include examples (as above) and troubleshooting tips for common issues. We will also describe how to use the `tasks.json` entry points if applicable (e.g., "to add a new task or modify terrain parameters, edit tasks/tilled_soil.json with instructions given inline").

- **Anonymized Artifact Plan:** Since the codebase is private, for the submission we will prepare an **anonymous zip or repository** (e.g., on an institutional figshare or a temporary GitHub under pseudonym) containing the necessary code. This will be referenced in the paper as "attached as supplemental material" or via an anonymous link. The artifact will exclude any identifying comments or metadata. If using a container, we'll push it to a registry under a neutral name. We'll double-check that no author names or URLs are present in the config or code. Our plan is to make the code publicly available upon acceptance, but for review we will ensure the artifact is accessible and executes as claimed.

By checking off each of these items, we aim to meet or exceed reproducibility guidelines of major venues (e.g., NeurIPS/CoRL reproducibility checklists).

Risk Register and Mitigations

1. **Risk: Tight Timeline and Compute Constraints (Schedule Slip).** *With only 30 days and multiple experiments to run, there is a risk not all tasks finish in time or we encounter unexpected delays (e.g., a training run takes longer than anticipated or needs rerun).*
   Mitigation: We prioritized the most critical experiments (see top 3) to run first and in parallel when possible. We also plan for slack in Week 4 by finishing experiments by end of Week 3, leaving time for reruns if needed. If a certain idea's run is taking too long or a hyperparameter needs tuning, we'll cut scope (e.g., drop lower-priority experiments like entropy variation) to ensure the main results are solid. We will also leverage any existing logs/models to avoid retraining stages unnecessarily (for instance, reuse Stage-1 policy from earlier work to save time).

2. **Risk: Baseline or New Experiment Underperforms Unexpectedly.** *It's possible that an idea we expect to improve things (or a baseline behaves in a certain way) doesn't meet expectations – e.g., maybe the no-curriculum baseline does better than anticipated or the slip-penalty hurts performance, complicating our narrative.*
   Mitigation: Have a contingency interpretation for each outcome. Our analysis plan is flexible: even if, say, the no-curriculum baseline learns somewhat, we can instead focus on sample efficiency ("it achieved lower reward even if it eventually walked, needing 5x more steps"). If a proposed improvement (like Stage-3 pushes or slip penalty) doesn't yield a clear win, we can

report it transparently and frame it as an insightful finding (or move it to appendix to not dilute the main story). We will back every claim with evidence, so even "negative" results are valuable. The key is to maintain a *defensible contribution*: we won't promise any result we can't support by data by submission time.

3. **Risk: Integration Bugs or Simulator Issues.** *Because we rely on IsaacSim (which can be finicky with physics or version changes) and we're making code modifications (adding perturbations, new metrics), there's a risk of breaking the environment or encountering sim bugs (e.g., unstable contact simulations, crashes).*
   Mitigation: We will stick to a stable version of IsaacSim that we've been using (avoid last-minute upgrades). Any new feature (like applying forces or computing new metrics) will be tested in isolation first. For example, we'll write a small script to apply a push to the robot and observe its effect before integrating into training. We also plan to use **determinism mode and smaller time-steps** if needed to improve simulation stability. If a serious simulator issue arises (like inconsistent physics), we will document it and potentially simplify the experiment (e.g., reduce push magnitude or disable a problematic feature) rather than lose time. Additionally, keeping all random seeds allows us to reproduce crashes and debug quickly.

4. **Risk: Simulation-Only Results Not Convincing Enough.** *Since we do not have a real-world experiment or the originally intended plant obstacle environment, reviewers might critique that the work is "only in sim" or not novel.*
   Mitigation: We will emphasize the complexity and realism of our simulation (tilled soil is a deformable, high-friction terrain – not a trivial surface). We will also highlight that sim-only papers are acceptable at venues like CoRL/ICRA when they address hard control problems or present thorough analysis. To strengthen our contribution, we incorporate the disturbance robustness aspect (simulating unpredictable events) and draw parallels to real-world scenarios (e.g., farming robots on soft soil). In the limitations, we'll explicitly acknowledge lack of real-world validation, framing our work as an important stepping stone. We'll also ensure our literature review positions our work as new (for instance, if no prior paper specifically tackled **biped locomotion on soft tilled soil** with RL, we claim that novelty clearly). Including a video of the simulation results can also help convince reviewers of the qualitatively difficult behavior achieved.

5. **Risk: Reproducibility and Artifact Issues (Private Code).** *As the codebase is private and complex (IsaacSim, RL code, etc.), there's a risk that reviewers face difficulties running our code or that anonymity is broken inadvertently.*
   Mitigation: We invest time (as scheduled in Week 4 and Idea 10) to polish the artifact: test it on a clean setup, provide a Docker, and remove identifying info. We also plan to provide **extensive documentation** so that even if the code is complex, the steps to reproduce are straightforward (copy-paste commands). If certain proprietary components cannot be fully shared, we'll find substitutes or clearly explain how to obtain them (e.g., "download IsaacSim from NVIDIA" with link). By following the reproducibility checklist above, we reduce the chance of artifact issues. Moreover, we will have an internal colleague act as a "pseudo-reviewer" to run through our README and catch problems before submission.

(Each risk above is tracked throughout the project. We will review this register each week to ensure mitigations are in place or adjust the plan if a new risk emerges.)

Submission-Readiness Checklist

Before the final submission, we will verify the following to ensure the paper and artifact meet academic venue standards:

- **Results Presentation (Tables & Figures):** All key results are presented in clear tables or graphs. We have a main results table comparing our method to baselines on all metrics (falls, error, slip, force) with appropriate units (e.g., "falls per 100m"). Figures are high quality and annotated: e.g., a plot of learning curves over training steps for curriculum vs. no-curriculum, or a bar chart of performance metrics. We've double-checked that every figure and table is referenced in the text and properly labeled (with captions explaining them). Any figure showing the environment or robot (if included) is visually clear (high contrast, proper labels).

- **Appendix and Logs:** An appendix is prepared (if allowed by the venue) containing additional details that don't fit in main text. This may include hyperparameter tables (learning rates, network architecture, etc.), pseudocode of the training algorithm/curriculum, additional plots (like per-stage training curves), and perhaps excerpts of raw log files to demonstrate training stability. We will also include in the appendix or supplemental material some example **training log output** for a few iterations to illustrate what metrics we tracked (for transparency). The appendix will also state any compute details or parameter settings as required by the conference's reproducibility checklist.

- **Limitations Section:** We have a candid **limitations** discussion. Here we will mention that *our results are in simulation only* and that we did not model plant interactions or vision-based terrain sensing. We'll note that the policy hasn't been tested on real hardware, so real-world transfer is an open question. Additionally, any known failure modes (e.g., maybe the policy struggles if the soil is too soft or if slopes are introduced) will be briefly acknowledged. By stating these, we show awareness and preempt reviewer criticisms. We'll position them as avenues for future work (e.g., "In future, incorporating deformable objects (plants) or performing sim-to-real transfer would be important next steps.").

- **Ethics & Broader Impact:** Although this is primarily a robotics learning paper, we will include a brief statement on societal impacts if required (some venues have this as a checklist item). We'll note that improved robotic locomotion in agriculture or uneven terrain has positive impact (robots can perform disaster-rescue or farming tasks in hazardous or tedious conditions, potentially reducing risk to humans). We'll also mention any negative impacts or ethical considerations: for example, ensuring that testing on real robots is done safely (so as not to damage the robot or environment, though in our case we didn't do real tests). We might note that there are minimal ethical concerns in simulation research itself, aside from the general issue of energy consumption for training. If relevant, we could mention that a field-deployed "Hunter" robot would need to not harm the environment (e.g., not crushing plants excessively – but since we don't simulate plants, we just flag that real deployment should consider that). Overall, this section will be short but shows we considered the broader context.

- **Compliance with Venue Format:** We will ensure the paper meets the target venue's formatting guidelines (page limit, reference style, etc.). For example, if targeting **CoRL or ICRA**, we'll use the IEEE or RL conference template, stay within the 6(+X) pages limit, and include all necessary sections (like related work, which we won't forget to write in final draft). All citations will be checked for completeness and no placeholder "[??]" references remain. The writing style will be adjusted to venue expectations (for CoRL/ICRA, a bit more emphasis on robotic significance; for NeurIPS, more on the ML novelty).

- **Supplementary Video:** *(Optional but recommended for robotics venues)* We will prepare a short video (1-2 minutes) demonstrating our results – e.g., the simulated biped walking on the tilled soil, possibly side-by-side comparisons of our policy vs. a baseline (like the flat-ground policy failing amusingly). While not strictly a "checklist" item, having a video can greatly enhance the submission. We will ensure the video is anonymized (no author names in captions) and upload it per submission guidelines. We'll mention in the paper that a video is provided.

- **Final Artifact Anonymity Check:** We'll do a final pass through the code and artifact to remove any hard-coded paths or metadata revealing author or institution. The submitted artifact will be named generically. We will also verify that the artifact does not violate any license (IsaacSim assets might have certain restrictions – if so, we will mention that the reviewer needs to agree to NVIDIA terms when installing IsaacSim, which is standard).

By verifying each of these items, we aim to submit a polished, professional paper package. Our goal is that a reviewer can clearly see our **contribution (robust bipedal locomotion via curriculum)**, trust our results due to the thorough evaluations and provided code, and appreciate the minimal-yet-meaningful scope of our work achieved in simulation.

Optional: Paper Outline & Figure Suggestions

(As an additional planning aid, here is a rough 1-page outline of the eventual paper with suggested figures.)

- **Title:** *Robust Bipedal Locomotion on Soft Terrain via Curriculum Reinforcement Learning* (tentative, highlighting terrain and curriculum)

- **Abstract:** (4-5 sentences summarizing problem, method (3-stage curriculum in sim), results (improved stability on tilled soil, disturbance robustness), and significance.)

- **1. Introduction:** Introduce the challenge of locomotion on compliant uneven terrain (like tilled soil, which can cause slippage and falls). Cite how prior work often did flat ground or needed special tricks. State that we use a 3-stage curriculum RL to tackle this, achieving stable walking in simulation. Emphasize "despite being simulation-only, our results are a step toward real-world field robots." End with a clear sentence on **contributions** (e.g., "(i) a curriculum training pipeline for soft terrain, (ii) demonstration of robust gait with improved metrics X, (iii) thorough evaluation and open-source code for reproducibility").

- **2. Related Work:** Cover relevant areas: curriculum learning in robotics[1], legged locomotion on rough/compliant terrain[5], maybe domain randomization and sim-to-real. Position that our work is unique in focusing on biped on deformable soil (most works are quadrupeds or rigid terrain) and in the specific curriculum design (if ours is unique with disturbance stage, mention that novelty).

- **3. Method:** Describe the **environment** (Hunter biped model, tilled soil terrain – describe how soil is modeled, e.g. heightfield or compliant contact parameters). Explain the **3-stage curriculum**: Stage-1 flat ground training (perhaps to get a basic gait), Stage-2 tilled soil training (the main challenge, possibly progressively increasing difficulty if we did that), Stage-3 random push perturbations (to improve robustness). Provide pseudocode or a diagram for curriculum progression. Describe the reward function (e.g., components for forward velocity, energy, maybe

slip penalty if used) and the policy architecture (e.g., 2-layer MLP, observation space details like proprioceptive state, no vision). Mention any special training details (PPO hyperparams, etc.).
Figure Suggestion 1: A schematic diagram of the curriculum: perhaps three panels showing (a) Stage1: robot on flat ground with maybe an arrow indicating guided training, (b) Stage2: robot on tilled soil terrain, (c) Stage3: an arrow pushing the robot. Arrows between them indicate progression. This visually communicates the training pipeline.

- **4. Experiments:** Break into sub-sections:

- **4.1 Training Performance:** Show learning curves (reward vs. steps) for our curriculum and maybe for the no-curriculum baseline to illustrate how curriculum stabilizes/improves training. Also mention training duration, etc.
  Figure Suggestion 2: Plot of average return or success rate over training steps for: curriculum vs. no-curr vs. maybe one other baseline. Perhaps curriculum reaches high performance faster or converges higher.
- **4.2 Evaluation on Tilled Soil:** Present the main quantitative results. E.g., a table of metrics: falls/100m, tracking error, slip rate, peak force for (i) our full method, (ii) no-curriculum, (iii) flat-only, (iv) smaller net (if space permits). Highlight in text that our method has the lowest falls and lowest slip, etc.
  Figure Suggestion 3: A bar chart or grouped bar for some key metric (like falls/100m for each method, to visually hammer home differences). Alternatively, a side-by-side screenshot of the biped: one where it's upright (our method) vs one where it falls (baseline) on the soil, as a figure, though that could also be a video snapshot.
- **4.3 Robustness to Disturbances:** If we did Stage-3 pushes, show results of that. For example, "with perturbation training, the robot withstands X N pushes with Y% success, whereas without it only Z% success." Possibly include a small demonstration figure.
  Figure Suggestion 4: Sequence of images (like a comic strip) of the robot being pushed and recovering vs. the one without training falling over. Or a plot of success rate vs. push force for with vs without perturb training.
- **4.4 Ablation Studies:** Summarize the findings from ablations like no randomization (perhaps "removing terrain randomization caused 30% higher tracking error on new soil types"), no curriculum stage, etc. This can be mostly text, or a small table if numeric. Ensure to cite relevant work to support findings (like [13] and others).

- **5. Discussion:** Interpret the results. For instance, why curriculum helped (perhaps the agent learns better footing on soft ground gradually), how the disturbance training could be analog to handling unexpected bumps or a future where plants provide lateral pushes. Discuss any observed behaviors (maybe the policy learned to lift feet higher to avoid dragging in soil, etc.). Mention the limits: e.g., policy might not generalize to completely different terrain like rocky ground, and we only did simulation. Possibly compare qualitatively to any real-world expectations or how one might transfer this (this is a good place for the limitation on no plants – e.g., "Our scenario lacked explicit obstacles like crops; a next step is to include those for a more comprehensive agricultural robot training.").

- **6. Conclusion:** Recap the contribution and key results (one sentence each). State that this work provides a foundation for robust bipedal walkers on deformable terrain, and perhaps that future work will tackle sim-to-real and more complex environment (closing on a positive, forward-looking note).

- **Appendix:** (If needed/allowed) Extra graphs, hyperparameters, maybe a figure of the simulation environment setup (if not in main text). For example, an actual image of the simulated tilled soil with the robot could be here if space is an issue in main text. We might include an example config snippet to show how easy it is to reproduce (for the artifact).

This outline ensures we cover all requested content in a logical flow. The figure suggestions aim to provide visual evidence of our claims: one showing the curriculum concept, others showing results comparisons, which will make the paper easy to digest. By following this structure, we align with typical conference expectations (Introduction -> Related Work -> Method -> Experiments -> Conclusion) and ensure all our minimal-scope contributions are highlighted.

---

[1] [4] [2010.03848] Guided Curriculum Learning for Walking Over Complex Terrain

https://arxiv.org/abs/2010.03848

[2] [3] [5] [6] Robust Humanoid Walking on Compliant and Uneven Terrain with Deep Reinforcement Learning

https://arxiv.org/html/2504.13619v1

[7] Available Environments — Isaac Lab Documentation

https://isaac-sim.github.io/IsaacLab/main/source/overview/environments.html