Multimodal Model-Based Reinforcement Learning for Autonomous Racing

Elena Shrestha
eshresco@umich.edu
Michigan Institute for Data & AI in Society
University of Michigan

Hanxi Wan
wanhanxi@umich.edu
Department of Robotics
University of Michigan

Chetan Reddy
chereddy@umich.edu
Department of Robotics
University of Michigan

Yulun Zhuang
yulunz@umich.edu
Department of Robotics
University of Michigan

Ram Vasudevan
ramv@umich.edu
Department of Robotics
University of Michigan

Abstract

Model-based reinforcement learning (MBRL) techniques have recently yielded promising results for real-world autonomous racing using high-dimensional observations. MBRL agents solve long-horizon tasks by building a world model and planning actions by latent imagination. This approach involves explicitly learning a model of the system dynamics and using it to learn the optimal policy for continuous control over multiple timesteps. As a result, agents may converge to sub-optimal policies if the world model is inaccurate. This paper proposes Lucid Dreamer, a end-to-end multimodal MBRL agent that leverages egocentric LiDAR and RGB camera observations through self-supervised sensor fusion. The zero-shot performance of MBRL agents is empirically evaluated on a 1:10 scale rover in simulation for unseen racing conditions and in a real-world environment to demonstrate sim-to-real transfer. Although only trained against five static obstacles in simulation, Lucid Dreamer safely avoided collisions with a dynamic rule-based agent in a zero-shot manner. This paper illustrates that multimodal perception improves robustness of the world model without requiring additional training data.

1 Introduction

Developing autonomous agents that learn to generalize beyond training data and adapt to novel and unseen environments remains a critical challenge. It is further exacerbated for autonomous racing where agents operate at the edge of their capabilities, making quick and precise decisions under pressure. They must accurately perceive the environment, accounting for both the track layout and behavior of other agents, and rapidly take actions that minimize lap times while avoiding collisions. Additionally, modeling errors from uncertainties in vehicle dynamics and noisy sensor measurements reduce the efficacy of conventional planning and control algorithms, necessitating more robust and adaptive approaches.

In recent years, learning-based control approaches that combine data-driven techniques with control theory have been successfully adopted for autonomous driving and related navigation tasks (Moerland et al., 2023; Evans et al., 2024). Among these, model-free reinforcement learning (RL) has gained prominence due to its ability to learn complex behaviors directly from interaction with the environment without requiring an explicit model of the system dynamics. However, model-free RL typically requires a vast amount of training data which is difficult to acquire for real-world applications where data collection is costly and potentially unsafe.

Instead of learning a policy that directly maps observations to actions, model-based RL methods first capture a reduced-order representation of the environment or system dynamics from high-dimensional observations (e.g., images). The world model is abstracted to a latent state space, which is a multi-dimensional space that encodes salient features of the environment relevant for decision-making. The policy is then learned end-to-end by mapping between the latent space representation and actions (Fig. 1). As a result, MBRL can be more sample-efficient. Because planning with the learned world model can illuminate factors influencing the agent's behavior, MBRL is more interpretable than model-free RL when using high-dimensional observations (Suzuki & Matsuo, 2022).

One of the key challenges in deploying MBRL agents is that the performance of the learned policy depends on the accuracy of the learned world model. To improve state estimation for autonomous racing, this paper explores methods for learning-based sensor fusion of egocentric 2D-LiDAR and RGB-camera observations. We extend the capability of Dreamer (Hafner et al., 2019; 2020; Wu et al., 2023), an existing state-of-the-art MBRL algorithm for visual control tasks that primarily relies on image observations, to leverage multiple modalities. We present a new multimodal MBRL agent called Lucid Dreamer that learns a hierarchical representation of the world model by fusing individual latent distributions of sensing modalities into an intermediate joint distribution using stacked encoders. To the best of the authors' knowledge, this is the first end-to-end implementation and demonstration of zero-shot transfer for a multi-agent setting of a multimodal MBRL agent. Multimodal perception is learned via self-supervised sensor fusion of high-dimensional LiDAR and camera observations collected from the racetrack. To evaluate the accuracy of the world model learned using various modalities, this paper qualitatively and quantitatively evaluates the robustness of the learned policy in both single-agent and multi-agent settings.

2 Related Work

Learning-based control for autonomous racing and related navigation tasks for unmanned ground vehicles (UGVs) has been an active area of research in the past few decades. We will provide a brief discussion of relevant RL and deep learning approaches in this section while referring readers to (Moerland et al., 2023; Evans et al., 2024) for a comprehensive survey.

Single Modality MBRL Brunnbauer et al. (2022) demonstrated sim-to-real transfer of single-agent autonomous racing using 2D-LiDAR rays instead of images for building the world model, and pre-training using expert demonstration from a state-of-the-art obstacle avoidance algorithm. Dwivedi et al. (2022) extended their work with a plan-assisted architecture that leverages planning in trajectory-space to improve exploration in single-agent races. However, both agents relied on a unimodal world model. In this work, we improve the accuracy of the world model by augmenting it with multimodal perception using 2D-LiDAR and RGB-camera, which provides richer and more diverse sensory information. We also demonstrate zero-shot transfer for multi-agent autonomous racing and showcase the agent's capability to adapt seamlessly to unseen scenarios without requiring additional training.

Multimodal World Model Tremblay et al. (2021) and Triest et al. (2022) improved the world model component of MBRL with multimodal perception for off-road driving using concatenated observations as input to the world model but did not demonstrate end-to-end learning. In this work, we focus on multimodal sensor fusion using hierarchical representations which improves perception of meaningful low-level features, outperforming simple concatenation for sensor fusion. In addition, we implement end-to-end learning and enable agents to operate in a multi-agent scenario.

3 Problem Definition

To capture the uncertainty in state estimation, the RL problem for autonomous racing is formulated as a Partially Observable Markov Decision Process (POMDP) defined by a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \Omega, \mathcal{O} \rangle$

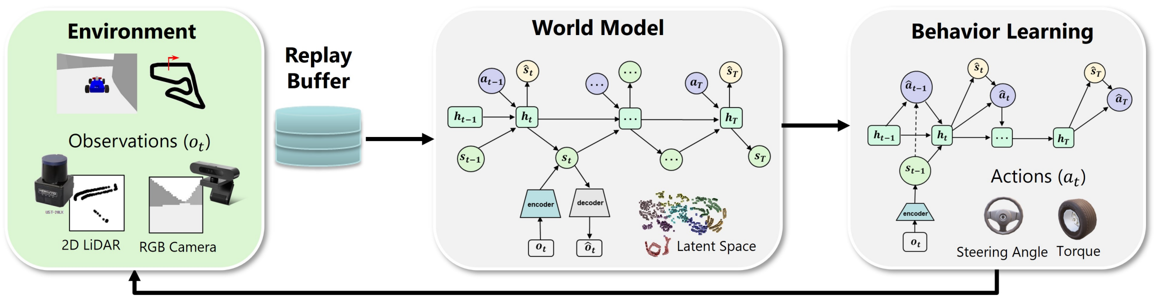


Figure 1: Overview of model-based RL components for autonomous racing. The environment provides observations from a 2D LiDAR and RGB camera. The world model encodes these observations into a latent space, updating the belief state through a recurrent model. The behavior learning component uses this information to generate actions (steering angle and torque) which are applied in the environment to collect new trajectories in the replay buffer.

containing the set of states ($s_t \in \mathcal{S}$), set of actions ($a_t \in \mathcal{A}$), the state transition function $\mathcal{T}(s_{t+1}|s_t, a_t)$ which characterizes the environment's dynamics, the reward function $\mathcal{R}(s_t, a_t, s_{t+1})$, set of observations ($o_t \in \Omega$), and the observation function $\mathcal{O}(s_{t+1}, a_t, o_t)$ (Sutton & Barto, 2018; Kaelbling et al., 1998). The observation function describes the probability of receiving an observation given the new state and the action taken. Given a trajectory $\tau = (s_0, o_0, a_0, r_0), \ldots, (s_T, o_T, a_T, r_T)$, the goal of the RL agent is to maximize the discounted cumulative reward $\mathbb{E}_\tau(\sum_{t=0}^{T} \gamma^t r_t)$, where γ is the discount factor and r_t is the reward from $\mathcal{R}$. In the context of autonomous racing, the agent tackles a continuous control problem, learning a policy that determines the optimal motor torque and steering angle to maximize progress on the track.

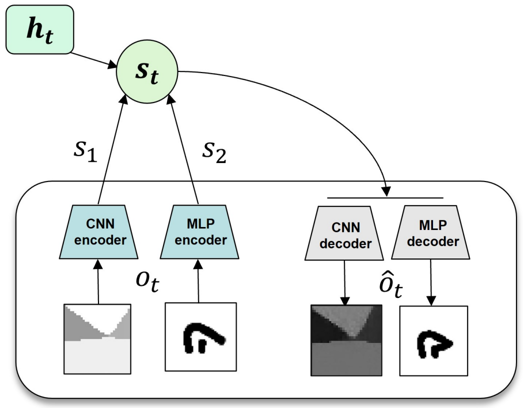
4 Method

Figure 1 highlights the key components of MBRL algorithms. During the world model learning phase, the agent learns the transition function that computes the belief state, or the distribution over the latent states, that capture the history of the environment's dynamics. Afterwards during the behavior learning phase, the agent learns a policy that maps the belief state to actions (e.g., motor torque and steering angle) that are then executed in the environment to collect new trajectories used for training.

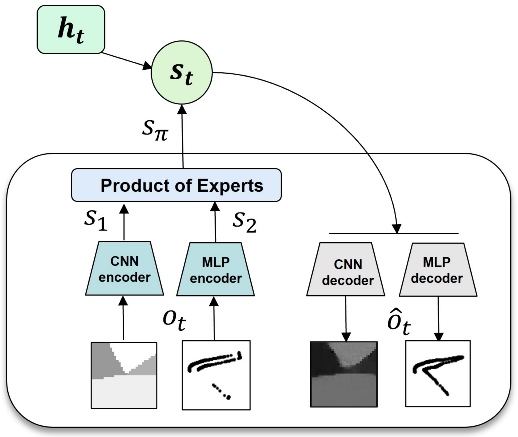
4.1 World Model

The world model is a combined latent state space consisting of compact representations of high-dimensional observations collected from multiple sensing modalities. In this work, the recurrent state-space model (RSSM) is extended to include a set of observations $O_t = (o_t^M, \ldots, o_T^M)$ where M represents the sensing modality (egocentric LiDAR or camera observations). RSSM learns the latent dynamics of the system through the representation, transition, observation, and reward models. These models are Gaussian with mean and variance parameterized by deep neural networks jointly updated by the parameter θ:

$$\begin{aligned} \textit{Representation}: &\ p_\theta(s_t|s_{t-1}, a_{t-1}, O_t) \backsim \mathcal{N}(\mu, \Sigma), && (1)\\ \textit{Recurrent}: &\ q_\theta(h_t|h_{t-1}, s_{t-}, a_{t-1}) \backsim \mathcal{N}(\mu, \Sigma), && (2)\\ \textit{Transition}: &\ q_\theta(s_t|s_{t-1}, a_{t-1}) \backsim \mathcal{N}(\mu, \sigma^2) && (3)\\ \textit{Observation}: &\ q_\theta(O_t|s_t) \backsim \mathcal{N}(\mu, \mathbb{I}), && (4)\\ \textit{Reward}: &\ q_\theta(r_t|s_t) \backsim \mathcal{N}(\mu, 1). && (5) \end{aligned}$$



(a) Simple concatenation.



(b) Lucid Dreamer (ours).

Figure 2: Multimodal sensor fusion architectures investigated in this work.

These models are jointly optimized to increase the evidence lower bound (ELBO). ELBO for the self-supervised Lucid Dreamer includes reconstruction terms for each of the sensing modalities, a reward loss, and a Kullback-Leibler (KL) divergence regularizer for the approximate posterior extended for multimodal perception:

$$\mathrm{ELBO}(O_t) = \mathbb{E}\Bigg(\underbrace{\ln q_\theta(r_t|s_t)}_{\text{Reward Prediction}} + \underbrace{\sum_t \beta_M \ln q_\theta(o_t^M|s_t)}_{\text{Reconstruction}} + \\ \underbrace{-\beta_{KL} D_{KL}\big(p_\theta(s_t|s_{t-1}, a_{t-1}, O_t)||q_\theta(s_t|s_{t-1}, a_{t-1})\big)}_{\text{KL Divergence Regularizer}} \Bigg). \quad (6)$$

The reconstruction losses are weighed by β_M, tuned to emphasize a particular modality, and the KL divergence regularizer D_{KL} is weighed by β_{KL}. The regularizer measures the difference between the representation and transition models, providing a learning signal that minimizes the information gain of the observations on the latent dynamics. Overall, the model parameters θ are jointly optimized to maximize the likelihood of observations and rewards for a given state visited during training.

Self-supervised Learning Lucid Dreamer relies on latent vectors encoded using a multilayer perceptron (MLP) encoder for the 2D LiDAR rays (1080 x 1) covering a 270° field-of-view (FOV), and a convolutional neural network (CNN) for the low resolution RGB camera images (64 x 64). Due to the difference in sensing modalities, observations are not processed through a single variational autoencoder as implemented in (Manderson et al., 2020) with top-down and forward-facing images. Instead, the observation model is implemented using a multimodal variational autoencoder (MVAE) where each modality is assumed to be conditionally independent (Wu & Goodman, 2018). One advantage of this sensor fusion technique is that the model can be robust to missing modalities based on the training scheme employed.

The belief state (s_t) is approximated using the product-of-experts (PoE) formulation (Cao & Fleet, 2014) with the representation model as the joint posterior and the transition model as the prior expert. Lucid Dreamer uses the PoE formulation to combine the observations into an intermediate latent space (z_π). Because all of the distributions are Gaussian, the PoE of the observations are analytically computed using the means and standard deviations of the individual modalities (Triest et al., 2022):

$$\prod_{o_t^M \in O_t} q(s_t|o_t^M) = \mathcal{N}\left(\frac{\sum_M \left(\frac{\mu_M}{\sigma_M}\right)}{\sum_M \left(\frac{1}{\sigma_M}\right)}, \mathbb{I}\left(\sum_M \frac{1}{\sigma_M}\right)\right). \tag{7}$$

The output of the PoE is then fused with the deterministic component (h_t) of the prior distribution using an encoder that parameterizes the posterior distribution (Fig. 2). Finally, the low-dimensional belief state is sampled from the encoded distribution. The stacked encoder design and sensor fusion using an intermediate latent state space enables Lucid Dreamer to learn a hierarchical representation of the environment. Without the intermediate latent space, the PoE formulation gives significant weight to high-dimensional observations, emphasizing images over LiDAR rays.

4.2 Behavior Learning

Given an encoded observation, the learned world model is used to generate imagined trajectories of states, rewards, and actions without reconstructing the observations (Fig. 1). In addition, a key advantage of Dreamer's actor-critic algorithm is that long-horizon behaviors are learned by backpropagating the value estimates through the latent dynamics, thereby achieving better gradient updates. The actor network learns a policy, $\pi_\phi(a_t|s_t)$, that aims to maximize the value estimates of the states while the critic network learns a value function, $v_\psi(s_t)$, that aims to match the value estimates, providing a learning signal for the actor network. Both the actor and value models use a dense neural network parameterized by ϕ and ψ, respectively. Additional information on the learning objective and equations used for value estimation can be found in Hafner et al. (2019).

$$r_t = \underbrace{100 * |p_t - p_{t-1}|}_{\text{Progress}} - \underbrace{0.1 * |\delta_t^{\text{steer}} - \delta_{t-1}^{\text{steer}}|}_{\text{Smooth Action}} + c_t \quad \text{where } c_t = \begin{cases} -1 & \text{if collision} \\ 0 & \text{otherwise} \end{cases}. \tag{8}$$

Reward Function The value estimate is a function of the reward function, which is pre-defined and deterministically calculated for a given state-action pair. The RL agent is trained using a dense reward function that maximizes progress on the track while avoiding collision (eq. 8). The normalized progress $p_t \in [0, 1]$ for the F1TENTH Gym is calculated for each pixel on the track using a distance transform applied to the grid (Brunnbauer et al., 2022). In order to deploy the model on hardware, the reward function also includes an action regularizer to smooth steering angles δ_t^{steer} learned by the actor-critic policy (Mysore et al., 2021; Seyde et al., 2021). Minimizing variations in steering angles intuitively leads to smoother trajectories and was also empirically shown to improve lap times in multi-agent training. Finally, a collision penalty is applied at the terminal state. Overall, the reward function encourages agents to learn the latent dynamics and avoid collisions with both the track walls and five static obstacles that are randomly initialized on the grid for the multi-agent setting.

Curriculum Learning The multi-agent experiments used curriculum learning with two stages: (1) agent first learns how to navigate the track without any obstacles present for 1M timesteps and then (2) agent learns to avoid collision with static obstacles (red cars) for another 1M timesteps. This curriculum approach ensures that the agent can consolidate its learning at each stage, ultimately resulting in a more robust and capable performance in real-world racing conditions. The trained agent then undergoes zero-shot evaluation against a dynamic obstacle, a rule-based agent using a conventional waypoint following algorithm. Waypoint Follower operates at a fixed speed with steering angle guided by a PID controller that tracks pre-defined waypoints set equidistant between the track's wall. The rule-based agent takes a deterministic path and functions as a dynamic obstacle for the MBRL agent.

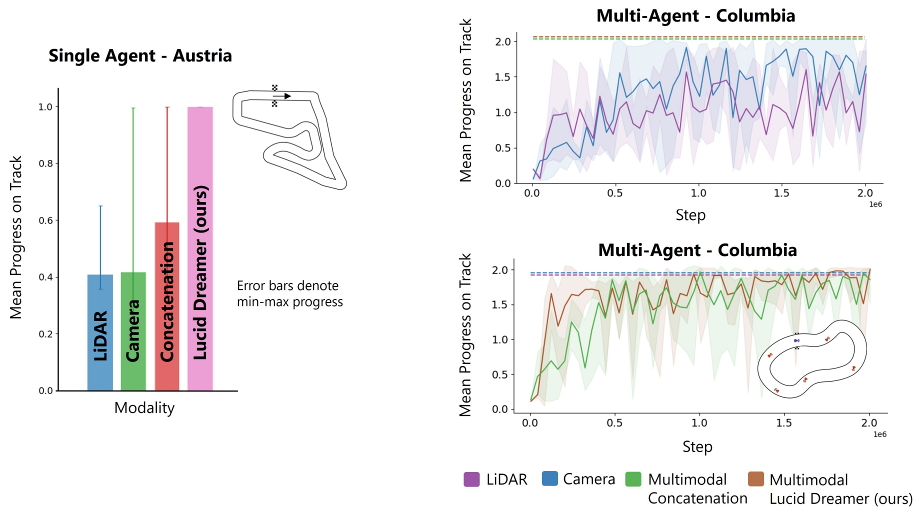


Figure 3: (left) Evaluation results on Austria when prediction horizon was reduced from H = 15 to H = 5. Bars denote mean progress over 30 episodes while delimiters show minimum and maximum progress. The best model for each agent was selected based on the mean progress from 5 seeded runs during training. (right) Multi-agent learning curves for single modality and multimodal agents on Columbia with static obstacles randomly initialized for each episode. Solid lines show the average mean progress from 5 seeded runs while dashed lines denote the highest mean progress achieved during training.

5 Experiments

Experiments were run using the F1TENTH Gym[1] which leverages the PyBullet physics engine for simulation. 2D-LiDAR scans with range of 15m and a 270° FOV were sampled at 25Hz. Low resolution images from a RGB camera with a 90° FOV were sampled at 100Hz. All modalities for the single-agent scenario were trained on Austria for 2M timesteps while agents in the multi-agent scenario were trained on Columbia for 2M timesteps. Position and orientation of the MBRL agent and static obstacles were updated at each environment reset in order to prevent overfitting to specific segments of the track and provide a diverse set of observations for representation learning. During evaluation, the rule-based agent is placed approximately 3m ahead of the MBRL agent on Columbia (minimum track width of 3.53m) in order to enable opportunities to overtake at various points along the track. Both agents are able to achieve top speeds up to 5m/s and have access to an identical range of control inputs with steering limited to ±24deg. Additional information on the training pipeline is provided in the Appendix.

5.1 Results

Figure 3 (left) shows evaluation results from Austria when prediction horizon was reduced from H = 15 to H = 5. All agents were able to complete the track after training on Austria for 2M timesteps, but Lucid Dreamer achieved the best mean progress and low variance. Visualization of the latent space is provided in the Appendix. The ability to identify distinctive features of the track appears to be more impactful for improving performance of the policy than achieving high-quality reconstructions in the F1TENTH Gym. Pure model-free agents were evaluated but did not complete the Austria track (Brunnbauer et al., 2022; Zhang et al., 2022), and therefore were excluded from benchmarks.

[1]https://github.com/CPS-TUWien/racing_dreamer

Method	Waypoint Follower	LiDAR	Camera	Multimodal Concatenation	Multimodal Lucid Dreamer (ours)
Mean Lap Time (s)*	22.54	28.87	**21.20**	21.37	21.64
Best Lap Time (s)	22.54	27.91	**20.53**	20.91	20.85
Mean Reward	-	42.13	65.97	82.23	**86.96**
Race Wins (%)	-	0.00	43.33	96.97	**100.00**
Collision (%)	-	80.00	53.33	3.33	**0.00**

*Mean calculation ignores incomplete trials

Table 1: Multi-agent zero-shot evaluation (head-to-head racing on Columbia: 30 Trials, 1 Lap each). Agents were trained against static obstacles but evaluated against a dynamic obstacle, a rule-based Waypoint Follower agent.

Impact of Multimodal Perception. Figure 3 (right) shows the learning curves (mean progress) for self-supervised single modality (LiDAR distance and camera) and multimodal agents on Columbia with five static obstacles randomly placed around the track. Both the LiDAR and camera-based agents converged to sub-optimal policies and on average completed less than two laps in 40s. The camera-based agent achieved a higher mean progress because it is able to discern between the static obstacles (red car) and the track walls. In comparison, both multimodal agents completed an average of two laps in 40s while Lucid Dreamer achieved a faster convergence rate.

Table 1 summarizes the zero-shot evaluation results of the head-to-head race against a rule-based agent (Waypoint Follower). Out of 30 episodes, LiDAR dreamer had the highest collision rate and the slowest mean lap times of the completed races. The latent space representation of depth-based observations may not be able to discern between a dynamic obstacle and the track walls due to insufficient training data. For example, the LiDAR observation of a tight corner could potentially be similar to when the dynamic obstacle is positioned directly ahead of the MBRL agent.

On the other hand, camera Dreamer achieved the best lap times on the track but remarkably only won 43.33% of the races. Of the remaining races, 53.33% prematurely ended due to collisions. The contrastive result highlights the delicate trade-off between maximising for speed and safety in head-to-head autonomous racing. While camera observations provide a higher spatial resolution of the environment, the FOV is limited to only 90° in front the agent. As a result, camera Dreamer is not as reactive as LiDAR Dreamer and will rigidly follow a racing line without adapting to the presence of the other agent on the track.

Impact of an Intermediate Latent Space. Because multimodal perception leverages strengths of LiDAR and camera observations, both multimodal agents outperformed their single modality counterparts. The combination of multimodal perception and collision penalty in the reward function enabled the agent to optimize for speed and safety. As previously discussed, the intermediate latent space improves perception of meaningful low-level features. As a result, Lucid Dreamer avoided collisions without employing an explicit safety layer over the policy.

5.2 Real-World Deployment

Domain Randomization For real-world deployment of the 1:10 scale rover on a custom track, significant adaptations were necessary to successfully complete the track in the single-agent setting. One critical technique implemented was domain randomization, which involves varying sensor noise (e.g., LiDAR accuracy, pixel distortion, etc.) and vehicle configuration (e.g., dimensions, inertial properties, etc.). This approach helps the model generalize better by exposing it to a wide range of scenarios and improves tolerance to discrepancies in conditions between the simulation and real-world environment.

Sensing Modality The addition of a proprioceptive sensor (i.e., IMU) was critical for successful sim-to-real transfer since it provided the agent with inertial measurements, enabling more accurate state estimation and better control in the real-world environment. While laser scans and RGB images offered environmental data, the onboard inertial measurements contributed essential information on vehicle dynamics, such as speed, ensuring a more accurate and robust performance. Laser scans and inertial measurements were combined using the PoE formulation.

Only the LiDAR and IMU-equipped agent successfully completed the track, as laser scans were more straightforward to adapt between simulation and the real world. Although RGB images contain detailed color information, their sensitivity to variations in lighting and shadows hindered the sim-to-real transfer for the camera-only agent. To maintain consistency with the simulator settings, the agent utilized segmented images, which offered a simplified and consistent representation by categorizing pixels into predefined classes such as track floor, walls, and obstacles. This abstraction reduced noise and variability in the input data, enabling more reliable and efficient processing. Segmented images were trained using the UNet architecture; however, the computational demands resulted in collisions due to the delayed update rate. The slower processing rate of the segmented images prevented timely updates, causing the agent to react too late to changes in the environment. Future work will focus on optimizing the segmentation model for real-time processing and exploring offboard model execution to mitigate these issues and improve the agent's performance in real-world scenarios.

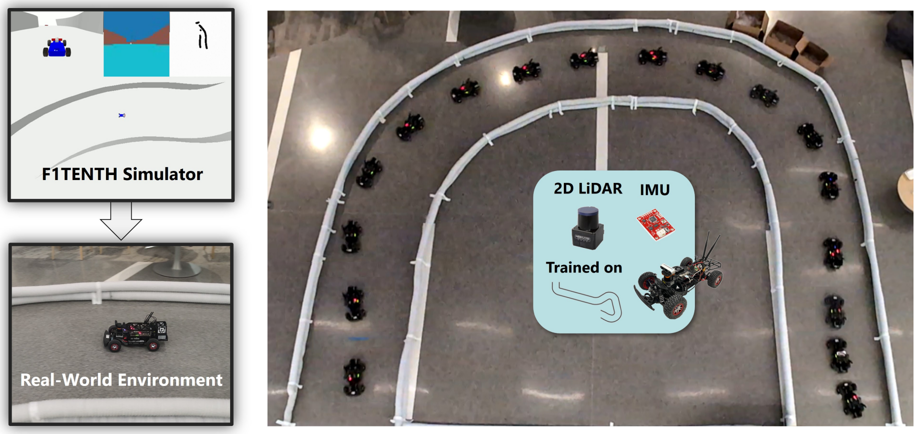


Figure 4: Real-world deployment demonstrating sim-to-real transfer using a custom track and a 1:10 scale rover equipped with 2D LiDAR and IMU. Agent was trained in simulator using a different track and successfully completed the real-world track in a zero-shot manner.

6 Conclusion

In this work, we showed that multimodal perception improves robustness of the world model and enables model-based reinforcement learning agents to safely avoid collisions while minimizing lap times in zero-shot head-to-head autonomous racing. Our proposed method, Lucid Dreamer, learns a joint representation of 2D-LiDAR rays and high-dimensional images in the latent state space using the product-of-experts formulation. Instead of simply concatenating all observations, Lucid Dreamer first encodes observations into an intermediate latent space before further encoding the learned representation into a belief state. Although only trained against five static obstacles, Lucid Dreamer safely avoided collisions with a dynamic rule-based agent in a zero-shot manner. Zero-shot head-to-head racing performance suggests that the ability to identify meaningful low-level features is more impactful in improving performance of the policy than achieving high-quality reconstructions. Finally, we conducted real-world experiments with a 1:10 scale rover on a custom track and demonstrated that multimodal perception enables sim-to-real transfer.

References

A. Brunnbauer et al. “latent imagination facilitates zero-shot transfer in autonomous racing, ” *International Conference on Robotics and Automation (ICRA)* , philadelphia, PA, USA. 10.: 7513–7520, May 2022.

Y. Cao and D. J. Fleet. “generalized product of experts for automatic and principled fusion of gaussian process predictions. ” *arXiv preprint arXiv:1410*, 7827, 2014.

T. Dwivedi, T. Betz, F. Sauerbeck, P. Manivannan, and M. Lienkamp. Continuous Control of Autonomous Vehicles using Plan-assisted Deep Reinforcement Learning. *2022 22nd International Conference on Control, Automation and Systems (ICCAS), Jeju, Korea, Republic of*, 10.:244–250, 2022.

Benjamin David Evans, Raphael Trumpp, Marco Caccamo, Felix Jahncke, Johannes Betz, Hendrik Willem Jordaan, and Herman Arnold Engelbrecht. Unifying f1tenth autonomous racing: Survey, methods and benchmarks, 2024.

D. Hafner, T. Lillicrap, J. Ba, and M. Norouzi. *“Dream to control: Learning behaviors by latent imagination, ” International Conference on Learning Representations.* 2019.

D. Hafner, T. Lillicrap, M. Norouzi, and J. Ba. *“Mastering Atari with Discrete World Models, ” International Conference on Learning Representations.* 2020.

L. P. Kaelbling, M. L. Littman, and A. R. Cassandra. “planning and acting in partially observable stochastic domains, ” Artificial Intelligence. 101:99–134, May 1998.

L. Maaten and Hinton G. Visualizing Data using t-SNE. *Journal of Machine Learning Research*, 9: 2579–2605, 2016.

T. Manderson, S. Wapnick, D. Meger, and G. Dudek. “learning to drive off road on smooth terrain in unstructured environments using an on-board camera and sparse aerial images, ” *2020 IEEE International Conference on Robotics and Automation (ICRA)* , paris, france. 10.:1263–1269, May 2020.

T. Moerland, J. Broekens, A. Plaat, and C. Jonker. *Model-based Reinforcement Learning: A Survey.* 2023.

S. Mysore, B. Mabsout, R. Mancuso, and K. Saenko. Regularizing action policies for smooth control with reinforcement learning. *2021 IEEE International Conference on Robotics and Automation (ICRA), Xi’an, China*, 10.:1810–1816, 2021.

T. Seyde et al. “is bang-bang control all you need? Solving continuous control with bernoulli policies, ” Advances in Neural Information Processing Systems. 34, 2021.

R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction.* 2nd Edition, A Bradford Book, 2018.

M. Suzuki and Y. Matsuo. “a survey of multimodal deep generative models, ” advanced robotics. 36:261–278, March 2022.

J. F. Tremblay, T. Manderson, A. Noca, G. Dudek, and D. Meger. Multimodal dynamics modeling for off-road autonomous vehicles. *2021 IEEE International Conference on Robotics and Automation (ICRA), Xi’an, China*, 10.:1796–1802, 2021.

S. Triest, M. Sivaprakasam, S. J. Wang, W. Wang, A. M. Johnson, and S. Scherer. “tartandrive: A large-scale dataset for learning off-road dynamics models, ” *2022 International Conference on Robotics and Automation (ICRA)* , philadelphia, PA, USA. 10.:2546–2552, May 2022.

M. Wu and N. Goodman. *“Multimodal Generative Models for Scalable Weakly-supervised Learning, ” Advances in Neural Information Processing Systems.* 2018.

P. Wu, A. Escontrela, D. Hafner, K. Goldberg, and P. Abbeel. “daydreamer: World models for physical robot learning. ” *Proceedings of the 6th Conference on Robot Learning, PMLR*, 205: 2226–2240, 2023.

R. Zhang, J. Hou, G. Chen, Z. Li, J. Chen, and A. Knoll. Residual policy learning facilitates efficient model-free autonomous racing. In *IEEE Robotics and Automation Letters*, volume 7, pp. 11625–11632. 10, doi, October 2022.

Appendix

Training Setting The prefill stage initializes the dataset with trajectories $\tau = (s_0, a_0, r_0), \ldots, (s_T, a_T, r_T)$ for 5,000 timesteps. All training runs use a random action strategy. Each training episode is terminated after 2000 simulation steps (20s) or automatically after a collision. Similarly, evaluation episodes are terminated after 4000 simulation steps (40s) or automatically after a collision. Distributed training for all agents was completed on a server with NVIDIA Tesla V100 SXM2 GPUs and Intel Xeon E5-2698 v4 CPUs while evaluation was performed on a laptop with a single NVIDIA RTX 3060 GPU and Intel i7-12700H CPU.

Hyperparameters Multiple intermediate latent space dimensions, $dim(z_\pi) = 200, 514, 1024,$ and 1080, were evaluated, with the latter yielding the best results. ELBO weights were set to $\beta_M = 1$ and $\beta_{KL} = 1$. Comprehensive tuning of hyperparameters is left to future work; only the action repeat ($AR = 4$) parameter was adjusted for the camera-based and multimodal agents from hyperparameters used in Brunnbauer et al. (2022) for LiDAR agents ($AR = 8$) due to discrepancies in the simulation update rate from using computationally expensive camera observations.

Latent Space Visualization Figure 5 shows the resulting latent space clustering for multimodal perception (left) and camera observations (right), colored with the corresponding position on the Austria race track. Observations were processed through a t-Distributed Stochastic Neighbor Embedding (t-SNE) (Maaten & G., 2016), a non-convex technique for converting high-dimensional Euclidean distances into conditional probabilities that represent similarities between neighboring data points. The resulting visualization is a two-dimensional map of high-dimensional inputs and preserves the number of data points before compression.

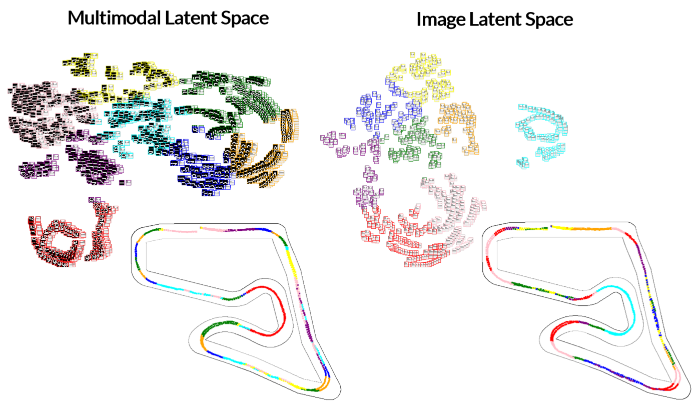


Figure 5: *left*) t-SNE of the combined latent space with camera and LiDAR observations (converted to occupancy grid for interpretability) overlapped and clustered with respect to the grid position on Austria. (*right*) t-SNE of the camera's latent space and corresponding grid. Multimodal latent space captured features that led to a finer segmentation of turns (3 clusters).

LiDAR rays were converted to occupancy grids to distinguish between data points within the clusters, but encoded in the combined latent space using the raw scans. The size of clusters and distance between clusters may not provide any meaningful information due to limitations in interpreting t-SNE plots. However, the segmentation quality of the race track can be evaluated because it elucidates the relative difference in clustering between multimodal and camera latent spaces. It is evident from the color-coded Austria track that the multimodal latent space captures additional features from the shared representation, which enable it to further segment turns on the grid using three clusters (blue, orange, green) compared to two (red, pink) in the camera's latent space. The segmentation quality improves performance of the behavior policy that maps latent features to actions.