



北京航空航天大學  
BEIHANG UNIVERSITY

# 无人机智能避障与导航

黄栋簷

北京航空航天大学计算机学院 23 级硕士研究生

2024 年 4 月 6 日

SafeDreamer Project 主要分为两个部分:

- ▶ RL 算法. SafeDreamer 结合了世界模型、拉格朗日约束和 CEEM 的一种基于安全强化学习的智能导航算法
- ▶ Safety-Gymnasium. Safety-Gymnasium 是一个基于 MuJoCo 引擎的仿真验证环境, 主要面向 SafeRL 算法的智能体仿真验证, 它支持单、多智能体; 基于 OpenGL 库, 支持丰富的视觉输入; 高度可定制化; 支持多种智能体和任务; 并且集成一个 SafeRL 算法库。

SafeDreamer 是 World Model, Lagrangian SafeRL methods 和 CCEM 三种方法的结合, 最终取得了良好的效果。

- ▶ **World Model:** World Model 是一种用来记忆和建模环境的神经网络模块, 在 SafeDreamer 中使用 DreamerV3 来作为 World Model, 其神经网络结构为 RSSM.
- ▶ **Lagrangian SafeRL methods:** Lagrangian SafeRL methods 是一种通过拉格朗日方法来求解强化学习中有约束优化问题的一种方法。在 SafeDreamer 中使用 Lag-aug 和 PID-Lag 来提高模型性能。
- ▶ **CCEM:** 约束交叉熵方法 (CCEM), 在每次迭代中, 首先从策略分布中取样, 选择一组精英样本策略, 并使用它们来更新策略分布。精英样本: 我们使用约束值对样本策略进行排序, 然后选择约束性能最好的策略。

1

SafeDreamer 由 world models 和 actor-critic models 组成.

3

Observation encoder:  $z_t \sim E_\phi(z_t | h_t, o_t)$

Observation decoder:  $\hat{o}_t \sim O_\phi(\hat{o}_t | s_t)$

Reward decoder:  $\hat{r}_t \sim R_\phi(\hat{r}_t | s_t)$

Cost decoder:  $\hat{c}_t \sim C_\phi(\hat{c}_t | s_t)$

Sequence model:  $h_t, \hat{z}_t = S_\phi(h_{t-1}, z_{t-1}, a_{t-1})$

Actor:  $a_t \sim \pi_\theta(a_t | s_t)$

Reward critic:  $\hat{v}_{r_t} \sim V_{\psi_r}(\hat{v}_{r_t} | s_t)$

Cost critic:  $\hat{v}_{c_t} \sim V_{\psi_c}(\hat{v}_{c_t} | s_t)$

15

## (a) World Model Learning

## (b) Actor Critic Learning

- ▶ **包含的模型:** SafeDreamer 包括世界模型和行动者-评论者模型。
- ▶ **处理观察和动作:** 在每个时间步骤  $t$ , 世界模型接收一个观察  $o_t$  和一个动作  $a_t$ , 并将观察压缩为一个离散的表示  $z_t$ 。
- ▶ **预测任务:** 这个离散的表示  $z_t$ , 连同动作, 被序列模型用来预测下一个表示  $\hat{z}_{t+1}$ 。
- ▶ **模型状态的定义:** 模型状态  $s_t = \{h_t, z_t\}$  由一个循环状态  $h_t$  和表示  $z_t$  的联接表示。
- ▶ **解码器的作用:** 解码器利用模型状态  $s_t$  预测观察, 奖励和成本。
- ▶ **行动者-评论者模型的输入:** 模型状态  $s_t$  充当行动者-评论者模型的输入, 以预测奖励值  $v_{r_t}$ , 成本值  $v_{c_t}$  和动作  $a_t$ 。

PID 拉格朗日方法是一种广泛应用于约束优化问题的算法。它的学习动态通常会展示振荡和超调，这在保证响应性和牺牲安全性方面是一个挑战。PID 拉格朗日方法利用了 PID 控制器的概念，使用比例、积分和微分这三个成分来平衡和控制学习动态，以优化安全性和效率。

## Algorithm 0: PID Lagrangian

**Input:** Proportional coefficient  $K_p$ , integral coefficient  $K_i$ , differential coefficient  $K_d$

Initialize previous integral item:  $I^0 \leftarrow 0$ ;

Initialize previous episode cost:  $J_C^0 \leftarrow 0$ ;

**while** iteration  $k$  continues **do**

    Receive cost  $J_C^k$ ;

$P \leftarrow J_C^k - d$ ;

$D \leftarrow \max(0, J_C^k - J_C^{k-1})$ ;

$I^k \leftarrow \max(0, I^{k-1} + D)$ ;

$\lambda_p \leftarrow \max(0, K_p P + K_i I^k + K_d D)$ ;

    Return Lagrangian multiplier  $\lambda_p$ ;

**end**

Constrained Cross-Entropy Method 是一种应用于安全强化学习问题的方法。该方法特点在于，它会显式地跟踪其在满足约束方面的表现。特别是，该方法在有限长度的轨道上定义约束，这是以预期的成本形式表示的。

### Algorithm 1: Constrained Cross-Entropy Method

**Input:** Initial parameters  $\theta_0$ , initial Lagrange multipliers  $\lambda_0$

Initialize parameters:  $\theta \leftarrow \theta_0$ , Lagrange multipliers:  $\lambda \leftarrow \lambda_0$ ;

**while** the stop criterion is not met **do**

    Generate a set of samples  $\mathcal{S}$  under the current model;

    Compute the utility values  $u_s$  and constraint violation  $c_s$  for all  $s \in \mathcal{S}$ ;

    Select a set of "elite" samples

$\mathcal{S}_{\text{elite}} = \{s \in \mathcal{S} : u_s \text{ is high and } c_s \text{ is low}\}$ ;

    Fit the model parameters  $\theta$  with  $\mathcal{S}_{\text{elite}}$ ;

    Update the Lagrange multipliers  $\lambda$  according to  $c_s$  and current  $\lambda$ ;

**end**

## 无人机智能避障与导航

1

黄栋旼

overview

RL 算法

Model Components

Preliminaries

Dreamer

Architecture

Safety-Gymnasium

intro

Agents

Tasks

Constraints

Vision-only tasks

SafePO

7

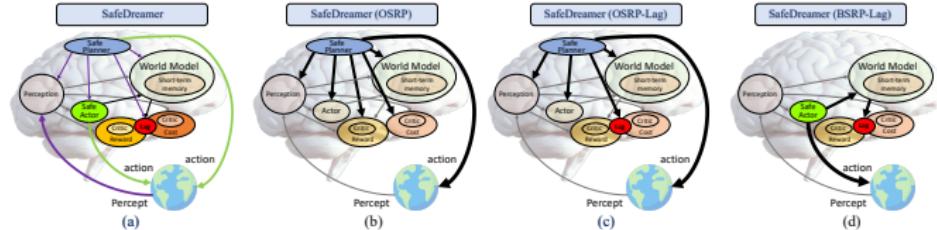


图 2: SafeDreamer Architecture

OSPR 和 BSPR 是 SafeDreamer 中的两种实现方式:

- ▶ OSRP: 每个决策时间点  $t$  都会进行在线规划过程, 该过程会在世界模型中从当前状态  $s_t$  生成状态-动作轨迹。每个轨迹都会通过学习到的奖励和成本模型以及它们的评价者进行评估, 从中选出最优的安全动作轨迹在环境中执行。
- ▶ BSPR: 在执行者训练期间, 我们在后台的世界模型中产生长度为  $T=15$  的想象的潜在回滚。我们从重播缓冲区的观察开始, 从执行者中采样动作, 并从世界模型中获取观察结果。

OSPR 在进行轨迹采样的每一步中, 会使用 Planner 生成 Action.

---

**Algorithm 2:** Online Safety-Reward Planning.

---

**Input:** current model state  $s_t$ , planning

horizon  $H$ ,

num sample/policy/safe trajectories

$N_{\pi_N}, N_{\pi_\theta}, N_s$ ,

Lagrangian multiplier  $\lambda_p$ , cost limit  $b$ ,  
 $\mu^0, \sigma^0$  for  $\mathcal{N}$

**for**  $j \leftarrow 1$  **to**  $J$  **do**

    Init. empty safe/candidate/elite actions set  $A_s, A_c, A_e$

    Sample  $N_{\pi_N} + N_{\pi_\theta}$  traj.

$\{s_i, a_i, s_{i+1}\}_{i=1}^{t+H}$  using  
 $\mathcal{N}(\mu^{j-1}, (\sigma^{j-1})^2 I)$ ,  $\pi_\theta$  within  $S_\phi$  with  $s_t$  as the initial state

    Select the top-k action trajectories with highest  $\Omega$  values among  $A_c$  as elite actions  $A_e$

$\mu^j, \sigma^j = \text{MEAN}(A_e), \text{STD}(A_e)$

**end**

**Return:**  $a \sim \mathcal{N}(\mu^J, (\sigma^J)^2 I)$

---

**Algorithm 3:** SafeDreamer. Color green is OSRP-Lag or OSRP, purple is BSRP-Lag

**Input:** batch length  $T$ , batch size  $B$ , episode length  $L$ , initial Lagrangian multiplier  $\lambda_p$

Initialize the world models parameters  $\phi$ , actor-critic parameters  $\theta$ ,  $V_{\psi_r}, V_{\psi_c}$

Initialize dataset  $\mathcal{D}$  using a random policy

**while** not converged **do**

    Sample  $B$  trajectories  $\{o_t, a_t, o_{t+1}, r_{t+1}, c_{t+1}\}_{t:t+T} \sim \mathcal{D}$

    Update the world models via minimize Equation (??)

    Condense  $o_{t:t+T}$  into  $s_{t:t+T}$  via world models

    Generate latent rollouts using the actor within the world model with  $s_{t:t+T}$  as the initial state

    Update the reward and cost critics via minimize Equation (??)

    Update the safe actor and  $\lambda_p$  via Equation (??) and Equation (??)

    // Only BSRP-Lag

$\mathcal{J}_C \leftarrow 0$

**for**  $t = 1$  to  $L$  **do**

        Condense  $o_t$  into latent state via  $s_t \sim S_\phi(o_t, s_{t-1}, a_{t-1})$

        Sample  $a_t \sim \pi_a(\cdot | s_t)$

            // Only BSRP-Lag

        Sample  $a_t \sim Planner(s_t, \lambda_p)$

            // Only OSRP-Lag or OSRP

        Execute action  $a_t$ , observe  $o_{t+1}, r_{t+1}, c_{t+1}$  returned from the environment

        Update dataset  $\mathcal{D} \leftarrow \mathcal{D} \cup \{o_t, a_t, o_{t+1}, r_{t+1}, c_{t+1}\}$

$\mathcal{J}_C \leftarrow \mathcal{J}_C + c_{t+1}$

**end**

    Use  $\mathcal{J}_C$  to update  $\lambda_p$  via Algorithm ??

            // Only OSRP-Lag

**end**

## 无人机智能避障与导航

黄栋旼

overview

RL 算法

Model Components

Preliminaries

Dreamer

Architecture

## Safety-Gymnasium

intro

Agents

Tasks

Constraints

Vision-only tasks

SafePO

1

**Safety-Gymnasium:** 这是一个专为安全强化学习 (SafeRL) 构建的仿真环境, 基于 Gymnasium 和 MuJoCo 构建. 通过增强现有的 Safety Gym 框架, 支持**vision-only**和多智能体任务. 此外, 该项目还集成了 SafePO, 一个单文件风格的算法库, 包含超过 16 种最新的算法。

10

## 主要组成部分:

inline=True

智能体模型, and-

安全任务

安全约束

图 3: Tasks from Safety-Gymnasium.

15

## 无人机智能避障与导航

1

黄栋旼

overview

RL 算法

Model Components

Preliminaries

Dreamer

Architecture

Safety-Gymnasium

intro

Agents

Tasks

Constraints

Vision-only tasks

SafePO

11

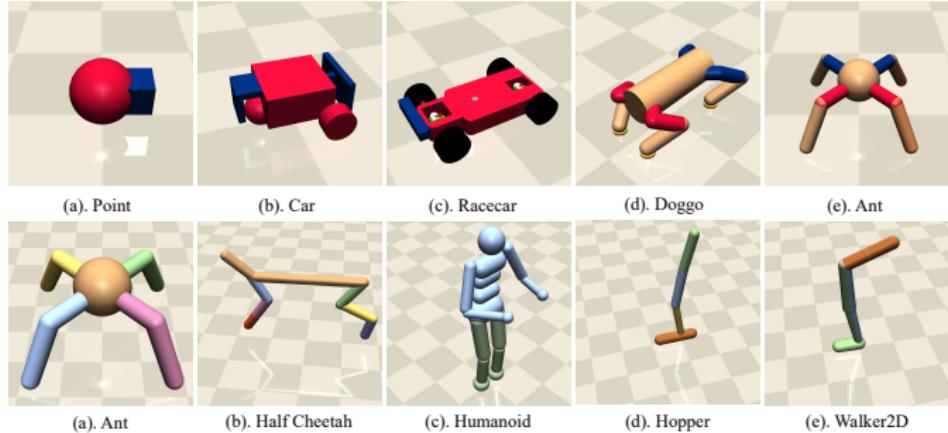


图 4: 上: 单智能体机器人。下: 多智能体机器人。

- ▶ **预设机器人:** Safety-Gymnasium 继承了 Safety Gym 里的三种现有智能体——Point, Car 和 Doggo。
- ▶ **优化改进:** 通过精心调整模型参数, 成功地缓解了 Point 和 Car 智能体在运行中过度振荡的问题。
- ▶ **新增机器人:** 在此基础上, 引入了两种额外的机器人——racecar 和 ant, 以丰富单智能体场景。
- ▶ **多智能体机器人:** 从多智能体 MuJoCo 中借鉴了一些配置, 解构了原本的单智能体结构, 并使多个智能体可以控制不同的身体部分。

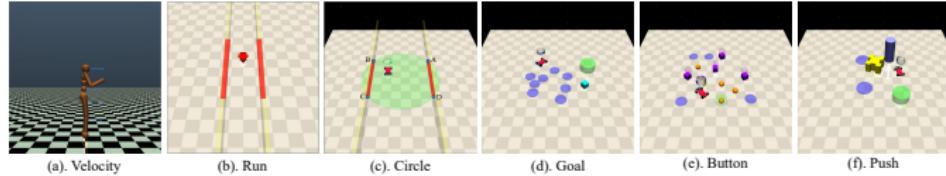


图 5: Tasks of Gymnasium-based Environments

- ▶ **Velocity:** 通过对铰链施加扭矩，使机器人在前向（右）方向上协调腿部运动。
- ▶ **Run:** 机器人从随机的初始方向和特定的初始速度开始，努力到达地图的另一边。
- ▶ **Circle:** 通过沿着绿色圆移动且不能进入红色区域外部来最大化奖励，因此，它的最优路径遵循 AD 和 BC 的线段。
- ▶ **Goal:** 机器人导航至多个目标位置。成功到达一个目标后，其位置在保持整体布局不变的情况下随机重置。
- ▶ **Push:** 目标是将一个盒子移动到一系列目标位置。与 Goal 任务类似，每个成就后会生成新的随机目标位置。
- ▶ **Button:** 目标是激活环境中分布的一系列目标按钮。智能体的目标是导航至并接触当前突出显示的按钮，称为目标按钮。

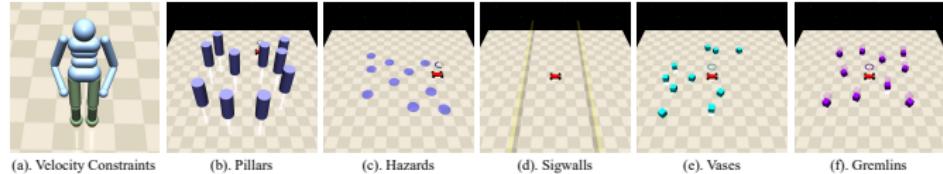


图 6: Constraints of Gymnasium-based Environments

- ▶ **Velocity-Constraint:** 在这些任务中，智能体通过更快地移动来获得更高的奖励，但也必须遵守速度限制以确保安全。
- ▶ **Pillars:** 用于在环境中表示大型圆柱形障碍。在一般情况下，接触柱子会产生代价。
- ▶ **Hazards:** 用于模拟环境中的危险区域，对智能体进入这些区域造成代价。
- ▶ **Sigwalls:** 开发的特殊用于 Circle 任务。从安全区内部越过围墙到外部会产生代价。
- ▶ **Vases:** 代表环境中的静态和易碎物体。触摸或移动这些物体会对智能体产生代价。
- ▶ **Gremlins:** 代表环境中可以与智能体互动的移动物体。

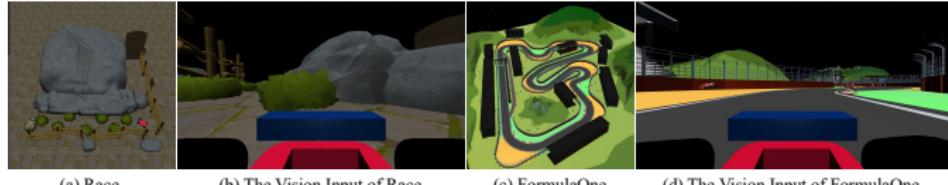


图 7: Vision-only Tasks of Gymnasium-based Environments.

## 原始框架的不足:

- ▶ **环境逼真度低:** 尽管 Safety Gym 的初始版本提供了基本的视觉输入支持, 但其环境的逼真性有待提高。Safety Gymnasium 的改进:

## Safety Gymnasium 的改进:

- ▶ **提高环境逼真度:** Safety Gymnasium 使用 MuJoCo 模型制定了一个更真实的视觉环境, 增强了环境的逼真性。
- ▶ **整合了更多的视觉输入:** 相较于原始框架, 这个改进版的环境能够整合 RGB 和 RGB-D 这两种输入, 从而提供了更为准确和全面的视觉信息。

## 无人机智能避障与导航

1

黄栋旼

overview

RL 算法

Model Components

Preliminaries

Dreamer

Architecture

Safety-Gymnasium

intro

Agents

Tasks

Constraints

Vision-only tasks

SafePO

15

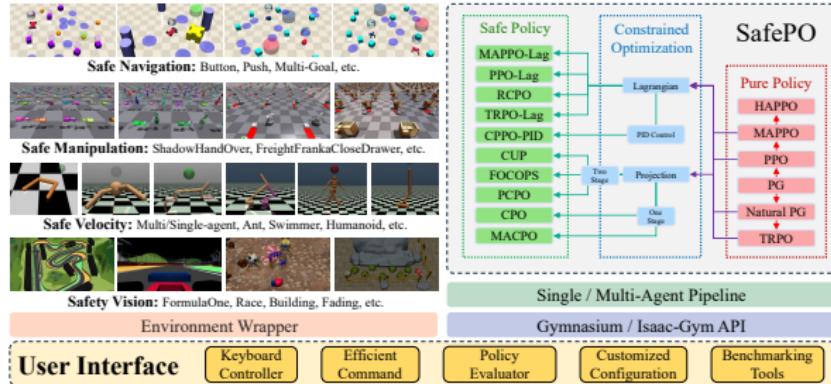


图 8: The Architecture of SafePO

- ▶ **正确性:** SafePO 确保了高度的准确性与可靠性，与现有基准相比具备超越或匹敌的性能。
- ▶ **可扩展性:** 新算法可以方便地集成到 SafePO 中，增强其可扩展性。
- ▶ **日志记录和可视化:** SafePO 支持 TensorBoard 和 WandB，提供了许多参数和结果的可视化，使训练过程透明化。
- ▶ **详细的文档指南:** SafePO 配有详尽的文档，从安装指南到高级定制都有涵盖，方便用户参考使用。