对使用强化学习的平流层气球自主导航的复现

赫英哲

摘要

要在平流层中有效地驾驭一个超压气球，需要考虑大量因素，如风速和太阳高度，而这一过程因预测误差和稀疏的风力测量而变得复杂。再加上需要实时做出决定，这些因素排除了使用传统的控制技术。在这里，我们使用强化学习[1]来创建高性能的飞行控制器。算法使用数据增强和自我修正设计来克服从不完美数据中强化学习的关键技术挑战，这已被证明是其应用于物理系统的主要障碍。本文根据论文 Autonomous navigation of stratospheric balloons using reinforcement learning[2]，对作者提供的超压气球强化学习模型进行复现，同时对数据结果进行评估。这些结果表明，强化学习是现实世界自主控制问题的有效解决方案。

关键词：强化学习；平流层风场；自主导航

1 引言

超压气球可以在平流层中自主运行数月，这使它们成为一个具有成本效益的通信、地球观测、收集气象数据和其他应用的平台。超压气球的高度是由其相对于周围大气的密度决定的。超压气球中，垂直运动是通过将气体抽入和抽出一个固定体积的包络来实现的，水平运动是由气球所在位置的风决定的。因此，为了导航，必须上升和下降以发现并遵循有利的气流。尽管看似简单，由于稀疏的风测量导致一种被称为部分可观测性的现象，一个好的控制策略必须权衡远端观察数据的成本和收益，长期的气球控制是具有挑战性的。

目前，我在我的导师和其他老师合作的“高空气球”项目中负责气球在平流层风场的控制问题。在经过一些调研之后，通过对比各种控制方法发现此论文中的强化学习方法明显优于其他传统控制方法。相比之下，过去关于自主导航的结果主要优化短期目标，而强化学习擅长于产生可以处理高维、异构输入和优化长期目标的控制策略。因此，复现此篇论非常适合我现在的情况。

希望通过对此篇论文的阅读和复现，能让我对项目有更深的理解，为未来的科研之路打好基础。

2 相关工作

2.1 控制器的评估指标

我们说一个气球在距离它的通讯站 50 公里以内是成功的，在这个距离内它可以完美的与地面设备通信。在距离站 50 公里范围内飞行时间的比例 (TWR50) 为评估控制器好坏的指标，TWR50 越高，表现越好。由于气球没有动力系统，只能通过充气或者放气改变超压气球密度来上升或者下降来寻找有利的风，控制过程如下图：

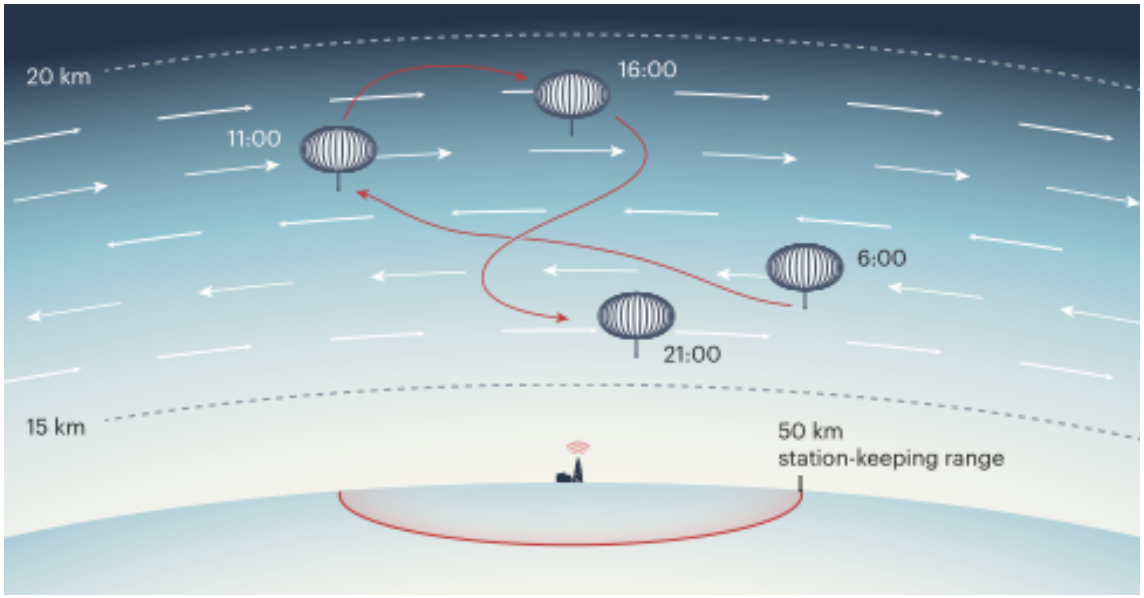

图 1: 气球控制示意图

2.2 控制器的控制方法

2.2.1 传统控制方法

传统控制方法 (StationSeeker), 在控制站范围外会追踪与站方向成锐角的风，有效地转向目的地。一旦气球进入范围，他就会寻找缓慢移动的风，使待在站内的时间更长。标准线性化技术可以实现短期的、低水平的气球控制。

2.2.2 强化学习控制方法

强化学习是智能体（Agent）以“试错”的方式进行学习，在与环境进行交互获得的奖励指导行为，目标是使智能体获得最大的奖励，通过接收环境对动作的奖励（反馈）获得学习信息并更新模型参数，如果 Agent 的某个行为策略导致环境正的奖励 (强化信号)，那么 Agent 以后产生这个行为策略的趋势便会加强。Agent 的目标是在每个离散状态发现最优策略以使期望的折扣奖赏和最大。强化学习特别适合处理操作少、环境简单、优化复杂的，长期目标的问题。通过计算奖励函数，让控制器找到一条最优控制策略

3 本文方法概述

3.1 风场环境的模拟

大量真实的训练数据是成功强化学习的关键。以前气球飞行的数据是不充分的，因为它们不能用来评估与历史行为的大偏差。另一方面，从物理大气模拟中产生足够精确的数据在计算上是不可能的。相反，我们基于 ECMWF 的 ERA5 全球再分析数据[3]创建了可信的风数据，该数据使用数值模型重新解释了历史天气观测数据。ERA5 提供了基线风，通过程序噪声进行修正，以生成高分辨率的风场。通过改变驱动程序噪声的随机种子，我们可以生成任意数量的场景并模拟预测错误。与数据增强相似可以提高强化学习控制器对建模差异的鲁棒性。风场环境模拟如下图：

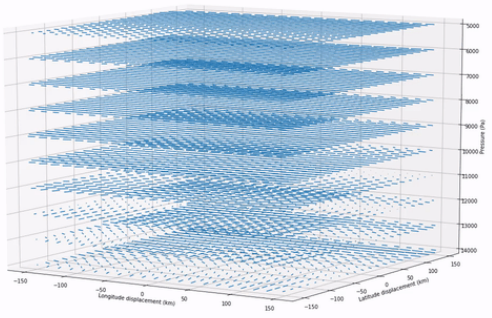

图 2: 模拟风场环境示意图

3.2 用神经网络估计动作价值函数

用 QR-DQN 算法[4]估计由奖励函数计算的动作价值函数[5]，用一个 7 层、600 个单元的神经网络对 QR-DQN 进行建模。

3.3 训练并评估结果

用相同的训练集分别用传统方法和强化学习方法训练，整理数据评估训练结果。

4 复现细节

4.1 算法的设计

4.1.1 奖励函数的设计

因为控制器以 TWR50 为评估指标，所以奖励函数的设计基于气球与通讯站距离：

$$r(\Delta) = 1.0(if \Delta < \rho) \tag{1}$$

$$r(\Delta) = c \times 2^{-(\Delta-\rho)/\tau}(otherwise) \tag{2}$$

其中 Δ 为通讯站的水平距离，ρ 为 50km。超参数 c 为惩罚因子，当气球在 50km 外时奖励值应有较大衰减。超参数 τ 为衰减率。取论文中的当 c=0.4,τ=100km 时提供最佳性能。此时奖励函数如下图：

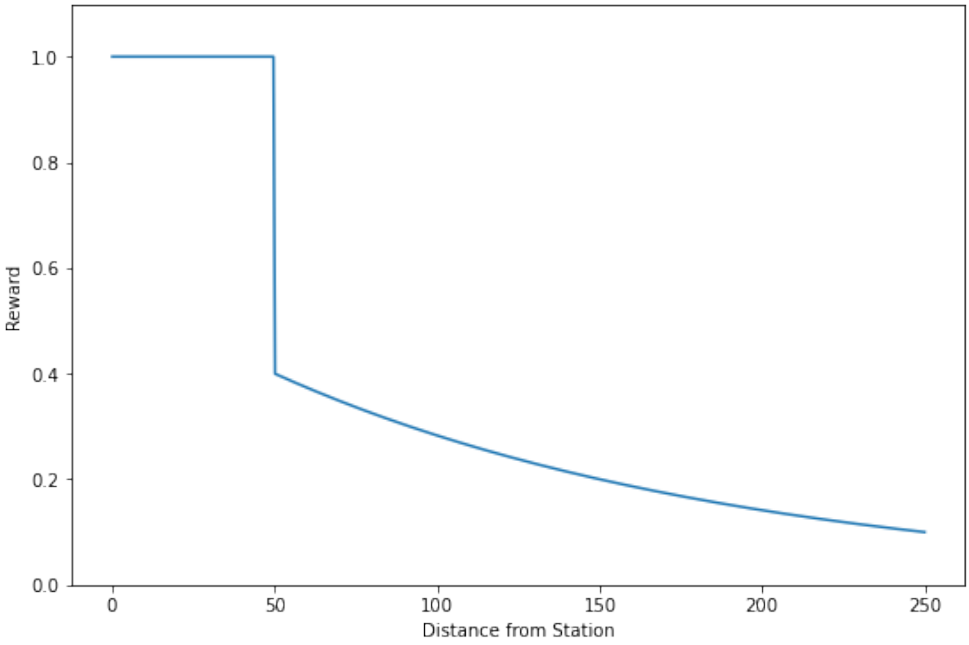

图 3: 奖励函数示意图

4.1.2 动作价值函数的计算

动作价值函数描述了在状态 s 下采取行动 a 所带来的预期回报：

$$R_{s,a} = E\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t | s_0 = s, a_0 = a)\right] \tag{3}$$

其中 s 为状态，指给定时刻的风场数据。a 为动作，指控制体的动作指令 (上升、下降或停留)。超参数 γ 为折扣率 (0<γ<1)，因为越是未来的奖励就不可靠，所以距离时间越远的奖励比重应该越低，论文中取 0.993。

4.1.3 用 QR-DQN 算法估计动作价值函数

用一个 7 层、600 个单元的神经网络估算动作价值函数。

$$R_{s,a} \approx \frac{1}{N} \sum_{i=1}^{N} \theta_{s,a^*}^i \tag{4}$$

该算法的开源实现可在网上获得[6]。

4.2 实验环境搭建

此论文的算法复现，环境搭建以及模型训练的相关软硬件配置如下：

软件：Ubuntu18.04 系统

- python3.8
- balloon learning environment == 1.0.1
- gym == 0.25.2
- jax == 0.3.25
- tensorflow-gpu == 2.7.0
- cuda == 11.7

- nvidia-driver == 515.65.01

硬件：

- GPU:NVIDIA GeForce GTX 3090

4.3 界面分析与使用说明

如果你只想运行此篇课程论文的复现结果，你只需要配置好上述环境后，运行上传在 Github 代码文件中的 blesh.DL.py 和 blesh.station_seeker.py 训练脚本，但是你需要在代码中修改生成的 checkpoint 和行动序列等文件的保存路径。我的训练脚本分别创建强化学习方法和传统方法的 Agent，同进行 1250 个二天的训练集训练然后进行 100 次评估, 评估不更新神经网络参数，只通过动作价值函数控制 Agent 运动后计算 Agent 在站内的时间比例。训练会生成 checkpoint 文件以便下次继续训练，还会生成动作序列和奖励信息等文件，在评估阶段生成的文件中，你可以看到每三分钟气球的飞行路径和每次评估的 TWR50。

如果你想创建你自己的 Agent，包括后续的训练以及评估，你可以按照以下步骤操作：

- 1、通过代码文件中的 agents.agent.py 接口创建你的 agent，或者由 agents.agent_registry.py 接口直接使用由论文作者提供的 agent。
- 2、通过调用代码文件中的 train_lib.py 的 run_training_loop 方法，设置文件保存路径，训练集大小和文件保存方式，写出你自己的训练脚本。
- 3、通过调用代码文件中的 eval_lib.py 的 eval_agent 方法，设置评估集的大小，保存返回的文件，写出你自己的评估脚本。
- 4、你可以通过设置 renderer=matplotlib 观看实时训练情况，包括气球与运动轨迹和电量信息，效果如下图：

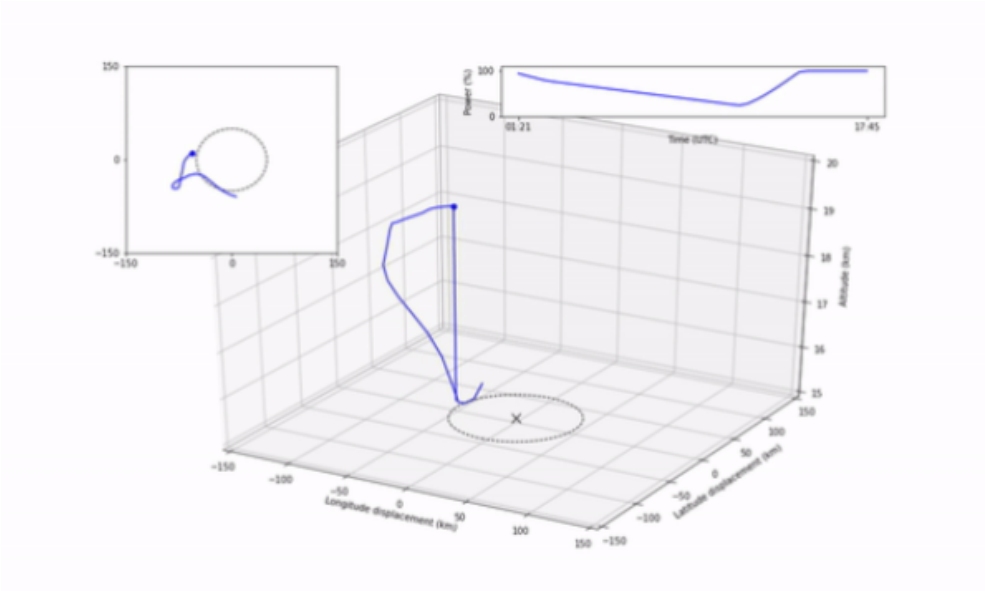

图 4: 实时训练示意图

4.4 我的改进尝试

奖励函数对强化学习的效果影响非常大，好的奖励函数不仅收敛快还能提高智能体性能的上限。我把公式 (2) 中当气球在 500km 外时奖励值设置为 0，理论上 agent 应该会获得更多的 TWR50, 因为 agent 在 50km 外不会获得奖励值而更多的飞行在站内范围。可实际上修改后的训练效果不如原论文，

在仔细阅读了论文后，发现在实际情况中风场很可能没有支持能让气球停在站内的风，将站外的奖励函数设置成递减形式，即使气球飞出站内也可以更多的让气球在靠近 50km 处徘徊，气球有更大的概率飞回站内。考虑到实际情况，明显将站外的奖励函数设计成递减比较合理。

5 实验结果分析

在用传统方法和强化学习方法分别进行 1250 个两天的训练集训练和 100 次评估后，在评估阶段生成的文本文件中 (已上传到 Github) 有气球的飞行路径信息和 TWR50。分别把 100 次评估的 TWR50 相加取平均值，求得强化学习方法 TWR50 平均为 35.6%，传统方法 TWR50 为 31.6%。数据方差较大，较符合本论文中强化学习方法表现优于传统方法，但是由于风场的不确定性因素较多，控制器每次的表现差距较大的结论。

6 总结与展望

在此次论文复现过程中，我根据此篇论文作者提供的方法理解了强化学习是如何工作和相较于传统算法的优势，并通过作者提供的训练模型对算法进行了学习与复现。

在对此篇论文的背景调研中，我发现强化学习方法的主流应用场景大多在棋类游戏 (如击败围棋冠军柯洁的 AlphaGo) 和电子游戏上, 因为此类场景操作简单确定性强，作者将强化学习应用在现实世界中是一次大胆的尝试。现实世界需要克服繁多的不确定性因素和难以从复杂环境中提取数据的缺点，作者通过设置更多的输入 (甚至把太阳高度角和风场预测值的不确定因素考虑在内) 试图完美的模拟现实情况来解决缺点一，通过历史风场数据加入随机噪声来生成更多训练数据解决缺点二，最后取得好的效果是一次巨大的成功。

在实际动手操作中，由于训练过程中真实的飞行路径的代价是在硬件上的缓慢模拟，串行的训练需要大量的时间和硬件资源，因为这些条件的限制我并没有达到论文中强化学习控制器在训练 24 天后达到的 55.1%TWR50。其次，论文中的许多理论和参数涉及到一些天文气象方面的知识，对这方面知识的缺乏也阻止我更好的理解作者的想法。

此后，我有更多的时间来训练出高性能的 Agent。但在机器上模拟和现实世界的情况肯定存在差距，可以在实际实验后把发现的问题在模拟时考虑周全，其次不同地区的风场由于季节和经纬度会有较大差异，通过调整奖励函数和修改茫多的超参数也是优化的方向。

参考文献

[1] SUTTON R S, BARTO A G. Reinforcement learning: an introduction, 2nd edn. Adaptive computation and machine learning series[Z]. 2018.

[2] BELLEMARE M G, CANDIDO S, CASTRO P S, et al. Autonomous navigation of stratospheric balloons using reinforcement learning[J]. Nature, 2020, 588(7836): 77-82.

[3] HERSBACH H, BELL B, BERRISFORD P, et al. The ERA5 global reanalysis[J]. Quarterly Journal of the Royal Meteorological Society, 2020, 146(730): 1999-2049.

[4] DABNEY W, ROWLAND M, BELLEMARE M, et al. Distributional reinforcement learning with quan-

tile regression[C]//Proceedings of the AAAI Conference on Artificial Intelligence: vol. 32: 1. 2018.

[5] WATKINS C J C H. Learning from delayed rewards[J]., 1989.

[6] CASTRO P S, MOITRA S, GELADA C, et al. Dopamine: A research framework for deep reinforcement learning[J]. arXiv preprint arXiv:1812.06110, 2018.