ROBOPIANIST: Dexterous Piano Playing with Deep Reinforcement Learning

摘要

在机器人手上实现像人一样的灵活性是机器人技术中的一个主要问题。强化学习近些年在这上面取得了显著的进展，是一个有前途的方法。但是与真正人类的能力相比，它通常能解决的任务范围很狭窄，也就是说它只能在非常有限的任务集上实现需要的灵活性。为了解决这个问题，这篇文章调研了让机器人手去弹钢琴的技能。弹钢琴是一种对人类来说都在挑战灵活性极限的技能，它需要非常高的时间上和空间上的精确性，还需要复杂的手指的协调和规划。这篇文章提出了 ROBOPIANIST，它能让模拟的人手学到多达 150 首钢琴曲而传统的基于模型的优化方法很难做到这一点。

关键词：高维控制；双手的灵巧操作；强化学习

1 引言

尽管关于在机器人手上复刻真实人类的手的灵巧性这方面的研究已经有数十年，高维控制仍然是个巨大的挑战。这一挑战激起了大量从机械设计角度和控制理论角度的研究。这其中基于学习的方法占据了主导地位，并在用手把玩各种形状的几何体上表现出了熟练的技能。然而这些任务相对与人类的能力来说就显得非常的具有局限性了。特别是，大多数任务都使用单个目标状态或者终止条件来给定非常具体的奖励或者损失函数，这限制了问题解决空间的复杂程度还经常为了达到目标状态而产生一些看起来不自然的行为。所以怎样赋予机器像人类一样的很高的运动控制的精度和灵巧性是一个仍待解决的大问题。

本文复现的工作为 2023 年发表于 CoRL 的文章：ROBOPIANIST: Dexterous Piano Playing with Deep Reinforcement Learning。在接下来的章节中，我首先会回顾将灵巧操作作为一个高维控制问题和机器人演奏钢琴的相关工作，然后阐述所复现的机器人弹钢琴的系统的具体工作，然后介绍我的复现工作，最后给出实验结果并作简单的总结。

在这个工作中，主要的考虑有：(i) 空间和时间上的精确性，(ii) 手指之间动作的协调性，以及 (iii) 动作规划。为此，研究人员构建了一个由两只机器人手 (位于一个完整的钢琴键盘之上) 构成的系统，其目标是演奏各种各样的钢琴曲，也就是根据乐谱 (形式为 MIDI)，在钢琴键盘上按照顺序按下一系列的键。如图 1。这双机器人手具有很高的自由度 (每只手有 22 个驱动器，总共有 44 个驱动器)，并且就像人手一样，部分是欠驱动的。控制这个系统需要对动作进行排序，以便机器手能够在正确的时间敲击正确的琴键；还需要能同时敲击不同的目标位置且手指之间相互不发生碰撞；还需要对如何按这些琴键进行规划，以便在时间和空间的限制下，机器手不仅在当前也能在之后的一段时间，按到需要按下的琴键。

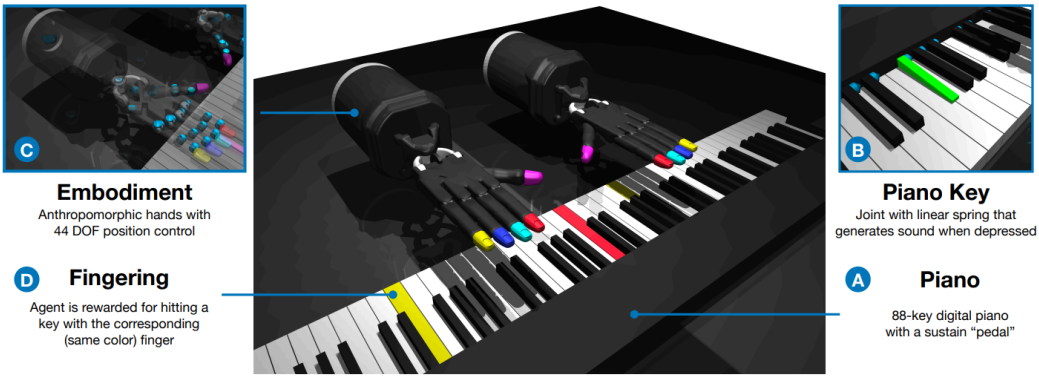

图 1. **ROBOPIANIST** 采用全尺寸数字键盘 (A)，带有 88 个建模为弹簧的按键 (B). 在钢琴演奏的任务中，两只拟人化的带有影子的手（左手和右手）(C) 的任务就是将一段编码为一条按键轨迹的钢琴曲演奏出来 (D)。

这篇工作提出了一个端到端的系统，ROBOPIANIST，利用深度强化学习的方法来合成能够在钢琴上演奏多种多样的曲目的策略。研究人员发现精心的系统设计与人类的先验 (以指法注释的形式) 的结合对它的表现至关重要。此外，该工作还引入了 ROBOPIANIST-REPERTOIRE-150，这是一个含有 150 首曲目的基准，这使得研究团队能够全面地评估本文提出的系统的表现。结果表明该系统的表现比最先进的 model-based 的方法提高了 83%（具体评估会解释这个 83%，见后文）。最后，该文章证明了多任务模仿学习在训练一个能够演奏多种曲目的统一策略的任务上是有效的。

2 相关工作

本文把相关工作主要分成两个领域：灵巧的高维控制和机器人钢琴演奏。

2.1 高维控制与灵巧操作

绝大多数控制相关的文献都使用比灵巧手低维很多的系统（例如单臂、简单的末端执行器)。即使是单手，也只有少数的泛化的策略优化方法被证明适用于高维度的灵巧手 [2] [3] [7] [11] [21] [6] [16] 它们中又只有一部分，在真实世界中展示出了结果 [2] [3] [7] [11] [21]。对于双手的结果就更少了，甚至只是在仿真环境里面也很少 [8] [4]。

这篇文章的一个非常突出的贡献是对 "任务成功" 的定义。例如，一般来说操作任务都被定义为在一个物体上施加连续的力和力矩，以实现想要的状态的改变（比如 SE(3) pose，即关节的位移和朝向，和速度)。这些任务对灵巧性的区分主要集中在机械臂的运动冗余（也就是可以有多种旋转使机械臂能够达到同一个目标位置）和末端执行器的复杂程度（从最简单的平行夹具到复杂度高的拟人手 [18] [8])。人们已经开发了多种方法来完成这样的任务，包括各种组合的 model-based 和 model-free 的强化学习、模仿学习、分层控制，等等 [23] [17] [7] [28] [22]。但是这些任务涉及到的灵巧程度是有限的 [13]，比如 re-orientation（也就是将对一种几何形状的操作迁移到另一种几何形状上)，relocation（也就是将物体放置到另一个位置)，操作简单的铰接的物体（如开关门）和使用一些简单的工具（如锤子、剪刀） [10] [26] [8] [11] [5]。尽管这些工作代表了一类重要的问题，这篇文章探索了对于灵巧性和任务成功的另外一种定义。

具体来说，对于上述的操作任务，目标状态是一些明确的，特定几何的函数；例如，开/关门，物体的 re-orientation，锤钉子等。这样的目标状态能够很大程度上减少针对具体每个任务的搜索空间。相比之下，ROBOPIANIST 包含更复杂的目标定义，通过音乐性的表现来进行编码。实际上，这变成了目标状态的高度组合可变序列，仅通过改变乐谱就能够扩展到任意的难度。“成功” 是根据整个 episode 的准确性来定义的，具体来说，是通过环境的时变的非解析的输出，也就是音乐。因此，这不是需要满足某种确定的终止/目标条件的最终状态问题（这种问题通常允许在除了最终状态以外的环节做出不太稳健的动作），而是贯穿整个过程的策略的行为，需要精确且具有音乐性。

相似的，关于 humanoid locomotion 或着更广泛的 “角色控制”（高维控制的另一个重要领域）的文献，主要是涉及到发现稳定的走/跑步态 [12] [25] [29]，或者对一个有限的全身运动先验集合进行蒸馏 [20] [14] [15]，用来在下游训练任务级的策略。任务成功通常是通过对动作的进度或者达到某个结束条件来定义的。有充分的证据表明，无休止地追求这些奖励可能会产生 “高奖励” 但是并不真实的行为。虽然 [20] [9] 等工作试图通过利用样例数据来捕捉风格目标，但是这些风格目标的奖励只是简单地附加到主要的任务目标之上。这种对多个目标的标准化产生了最优控制的任意主观帕累托曲线 (arbitrarily subjective curve)。相比之下，演奏一首音乐既需要客观可测的节奏和旋律的准确性，也需要主观的音乐感受的评估。数学上来说，这可以转化为风格约束满足，这为算法的创新开辟了道路。

2.2 机器人钢琴演奏

研究机器人弹钢琴的历史由来已久，一些工作致力于专门的硬件设计，还有一些致力于使用预编译的命令（开环）来合成定制的 controller 来弹一首曲子。 [24] 这篇工作使用一只机器手 (shadow hand [1]) 利用逆运动学和轨迹拼接来弹单个的琴键并回放一些简单的片段和一整首歌。更加新的研究 [?], 作者使用离线运动规划和逆运动学来模拟钢琴演奏，使用一个 7 个自由度的机械臂并使用基于迭代最近点的启发式方法来为一个四指的 Allegro 手选择指法。每只手都是单独模拟的，并且音频结果是后期组合的。最后，在 [30] 中，作者将弹钢琴定义为一个使用单手的四个手指的 Allergo 手 (Allergo hand) 在一个微缩版的钢琴上演奏的强化学习问题，并额外利用了触觉传感器作为反馈。但是这些工作考虑的演奏任务都相当的简单，比如最多可以弹六个连续音符或者三个连续和弦，而且每个和弦最多同时按下两个琴键。ROBOPIANIST 基准套件旨在通过提供按照难度分级的音乐作品课程，让通用的双手可控的 agent 能够像人类钢琴家一样弹琴技能可以日益增长。利用两只欠驱动的拟人手作为驱动器提供了一定程度的真实性，并暴露了掌握这类高维控制问题面临的挑战。

3 本文方法

本文的目标是使机器人能够展现出能够成功演奏具有挑战性的音乐作品的复杂的、高维度的控制能力。钢琴的大师需要：(i) 空间和时间上的精确性 (hitting the right notes, at the right time)，(ii) 协调性（能够同时按下多个目标按键，且手指之间不会发生相互的碰撞），(iii) 规划（当前琴键按下的方式或者说姿态要基于之后一段时间需要按下的那些琴键来考虑和设计）。如果我们仅仅通过在正确的时间按下正确的按键这样稀疏的奖励，这些行为是不会被学习到的。主要的挑战在于 exploration，高维控制使搜索空间非常大进一步加剧了这种挑战。

Reward	Formula	Weight	Explanation
Key Press	$0.5 \cdot g(\|\|k_s - k_g\|\|_2) + 0.5 \cdot (1 - \mathbf{1}_{\{\text{false positive}\}})$	1	Press the right keys and only the right keys
Energy Penalty	$\|\tau_{\text{joints}}\|^\top \|v_{\text{joints}}\|$	-5e-3	Minimize energy expenditure
Finger Close to Key	$g(\|\|p_f - p_k\|\|_2)$	1	Shaped reward to bring fingers to key

图 2. 用于训练 ROBOPIANIST agents 的奖励函数。τ 代表关节力矩，v 是关节速度，p_f 和 p_k 代表手指和琴键各自在世界坐标系的位置，k_s 和 k_g 分别代表琴键的当前状态和目标状态，g 是一个将距离转换成 $[0, 1]$ 之间的奖励的一个函数。

这篇文章通过一系列的系统设计和利用人类的先验来应对这种挑战。

3.1 人类先验

作者发现由于高维控制带来的搜索空间巨大，agent 很难仅仅通过一些稀疏的奖励函数的指引就学到弹钢琴的技能。为了克服这个问题，作者将指法标签纳入了奖励函数之中，如图 (2) 中的第三项 (Finger Close to Key)。当移除这个先验而只通过对按下需要按下的按键进行奖励时，即使经过大量的训练，agent 的能力也几乎没有进展。作者猜测这样的奖励函数不仅有助于 agent 按下当前琴键，还有助于 agent 实现在今后一段时间内按下目标琴键。由于默认情况下 MIDI 文件不提供指法标签，所以作者使用 PIG 数据集中的标注来创建了一个含有 150 首带指法标签曲目的语料库，以便在仿真环境中使用。

3.2 奖励设计

如图 2，作者首先包含了一个与应该按下的琴键被按下的程度成比例的奖励，然后针对不应该被按下的琴键如果被按下得足够的多以至于被激活的情况，添加一个恒定的惩罚项。这给了 agent 一些空间，可以将手指放在那些不应该被按下的琴键之上，只要这些琴键不被激活就可以。无论不应该被按下却被按下的键有多少，给个常数项的惩罚都很重要，否则 agent 会变得非常保守而悬停在钢琴上方不按下任何的琴键。也就是说 $0.5 \cdot (1 - 1_{falsepositive})$ 这个 reward 项，有鼓励 agent 去按下琴键，但是禁止 agent 把不应该按下的琴键按得非常深的作用。相比之下平滑的这个奖励项通过提供密集的学习信号，在探索应该按下的琴键的相关动作中发挥着很重要的作用。作者还引入了两个额外的项：(i) 鼓励手指在空间上接近它们需要按下的按键（根据上一节所描述的指法标签）来帮助探索，(ii) 最小化能量消耗，这可以减少 (不同随机数种子带来的) 方差，还能减少使用强化学习训练的策略容易产生的不稳定行为。在某个时间步，将上述几项加权求和就能得到总的奖励。

3.3 将未来纳入观察

通过在观察中添加未来的目标状态，agent 的性能得到了提高，也就是说，提高向前观察的时间区间到 L，如图 3。直观上，这使得策略能够更好地规划未来的音符，例如通过以一种能够使手指更及时地到达下一个需要按下的琴键的方式来放置手腕。

3.4 限制动作空间

为了进一步减轻在复杂搜索空间中进行 exploration 的复杂度，本文探索了在 Shadow Hand 上限制某些自由度 [27]，这些自由度要么不可能在人类的手上出现（例如两个小手指朝

Observations	Unit	Size
Hand and forearm joints	rad	52
Forearm Cartesian position	m	6
Piano key joints	rad	88
Active fingers	discrete	$L \cdot 10$
Piano key goal state	discrete	$L \cdot 88$

图 3. agent 的观察空间，L 代表向前观察的帧数

向相对)，要么对于绝大多数曲目来说是不必要的。实验还额外将大拇指的活动范围限制到一定程度。实验发现，虽然这样做很大地提高了学习的速度，但是通过更多的训练时间，完整的动作空间也能使得 agent 弹出相当的效果。

3.5 单任务的专家策略

使用强化学习对针对每一首曲目来训练一个策略。

3.6 多任务的统一策略

3.6.1 直接使用强化学习对多首曲目进行学习

由于搜索空间过于巨大，这个效果很差，几乎等于没有效果。

3.6.2 多任务的行为克隆

利用针对每个曲目学习到的专家策略，使用 Behavior Cloning 将这些策略蒸馏到同一个策略之中。但是实验效果仍然不好。由于效果不好，我没有做这个蒸馏的工作，具体做法和原因请参看原文，这里省略。

4 复现细节

4.1 与已有开源代码对比

该工作有源代码，源代码地址是：https://github.com/google-research/robopianist.git。但是源代码只给出了在 mac 和 linux 上运行实验的流程。由于我已经熟悉了在 windows 上使用 MuJoCo，所以我就尝试将代码迁移到 windows 上，经过一些工具的适配（比如音效工具）后，我成功地做到了，所以我合理推测，作者只是由于习惯而不是因为阻力没有给出 windows 的运行方法。我的代码位于：https://github.com/kikato2022/kikarobopianist.git。另外，我还修改了在强化学习过程中每一个 epoch 的训练范围，源代码在每一个 epoch，都要训练完一首完整的歌曲，我发现这样效率不高，所以我把训练范围改成了和前瞻范围一样的 L 帧，在使用双层循环，外层还是训练一整曲目，但是内层只会每次选取得分最低的 L 帧来进行训练，我发现经过这样使得训练速度变快，但是持续到 50 万次采样后，两种方式的最终效果是相当的。另外，我还发现，由于 menagerie hand 这个手部的模型手腕比人类粗，而且源代码没有在手腕上加上足够的旋转自由度，导致遇到需要两只手的手腕靠得非常近而两只手却相对远

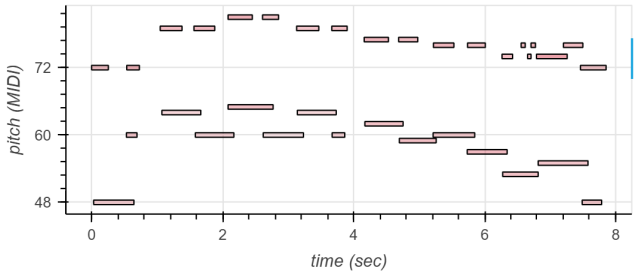

图 4. MIDI

的情况 agent 学习不到这种手法。所以我修改了原来的手部模型，给手腕加上了更大的自由度，然后这个问题得到了很大的改善。

4.2 实验环境搭建

使用开源的 MuJoCo 来构建整个钢琴演奏的仿真环境。钢琴是一个全尺寸的数字键盘，有 52 个白键和 36 个黑键，并使用 Kawai 手册作为亲琴键的位置和尺寸的参考。每个琴键都被建模为带有线性弹簧的关节，当按键在它的最大范围的 0.5° 以内时，被视为"active"，超过这个临界点，一个合成器就会生成该琴键对应的声音。同时还实现一种机制来延续当前任意活动音符的声音，用来模仿钢琴上延音踏板的机械效果。左手和右手都是来自 MuJoCo Menagerie 的 Shadow Dexterous Hand 模型，这个模型能够很好地建模人手的运动学。

使用 Musical Instrument Digital Interface(MIDI) 标准来将一段音乐表示为一个当前时间状态每个音符是打开或者关闭的时间序列，如图 4。然后再使用一个 message 来携带附加信息比如一个音符的音调和速度。这个轨迹用来作为 agent 的目标，告诉它在每个时间步应该有哪些琴键被按下。

使用 presision、recall、F1 score 来评估 agent 的熟练程度。presision 衡量的是一个琴键不应该被按下时，它是否没有被按下；recall 衡量的是一个琴键应该被按下时，它是否被按下了。F1 score 将 precision 和 recall 统一到一个指标里面，范围是 0(如果 recision 或者 recall 任意一个为 0) 到 1(precision 和 recall 都是 1)。主要使用 F1 score 作为的评估，因为它是音频信息检索中常见的评估方法。

将钢琴演奏建模为一个有限范围的马尔可夫决策过程 (MDP)，$(S, A, \rho, p, r, \gamma, H)$，其中 S 是状态空间，状态空间在前面已经提到过，如图 3，A 是动作空间，ρ 是初始状态分布，p 是 dynamics（对于仿真环境来说，dynamics 是确定的，所以不存在随机性问题）r 是 reward，$\gamma \in [0, 1)$ 是折扣因子，H 是范围。学习过程中 agent 的目标就是最大化 discounted reward 的期望，也就是 $E[\sum_{t=0}^{H} \gamma^t r(s_t, a_t)]$。agent 按照 20Hz 的频率来预测关节角度，然后将它作为 target 使用频率为 500Hz 的 PD controllers 将它转换为关节力矩。

4.3 训练细节

4.3.1 计算设备和运行时间

1 块 Nvidia 4090 GPU，训练一首歌曲的策略大概会花 8 个小时。一共训练了 32 首歌曲，陆陆续续训练了接近一个月的时间。

Hyperparameter	Value
Total train steps	5M
Optimizer	
Type	ADAM
Learning rate	3×10^{-4}
β_1	0.9
β_2	0.999
Critic	
Hidden units	256
Hidden layers	3
Non-linearity	`ReLU`
Dropout rate	0.01
Actor	
Hidden units	256
Hidden layers	3
Non-linearity	`ReLU`
Misc.	
Discount factor	0.99
Minibatch size	256
Replay period every	1 step
Eval period every	10000 step
Number of eval episodes	1
Replay buffer capacity	1M
Seed steps	5000
Critic target update frequency	1
Actor update frequency	1
Critic target EMA momentum (τ_Q)	0.005
Actor log std dev. bounds	$[-20, 2]$
Entropy temperature	1.0
Learnable temperature	True

图 5. 超参的设置

4.3.2 网络架构

对于强化学习中的 critic，使用 DroQ。每个 Q-function 都被参数化为一个 3 层的 MLP，使用 ReLu 作为激活函数。每层 layer 后都跟一个比率为 0.01 的 dropout 和层归一化。对于 actor，实现为一个 tanh-diagonal-Gaussian，同样使用 3 层的输出均值和方差的 MLP 进行实现。actor 和 critic 都有 256 个神经元的隐藏层，他们的权重使用 Xavier initialization 进行初始化，biases 被初始化为 0.

4.3.3 训练和评估

首先使用均匀分布收集 5000 个种子观测值，然后使用强化学习策略对动作进行采样。然后每次收到新的观察结果时，都执行一次梯度更新。使用 Adam 优化器来优化神经网络。每采样 10000 次后进行一次评估.

4.3.4 超参数的设置

如图 5

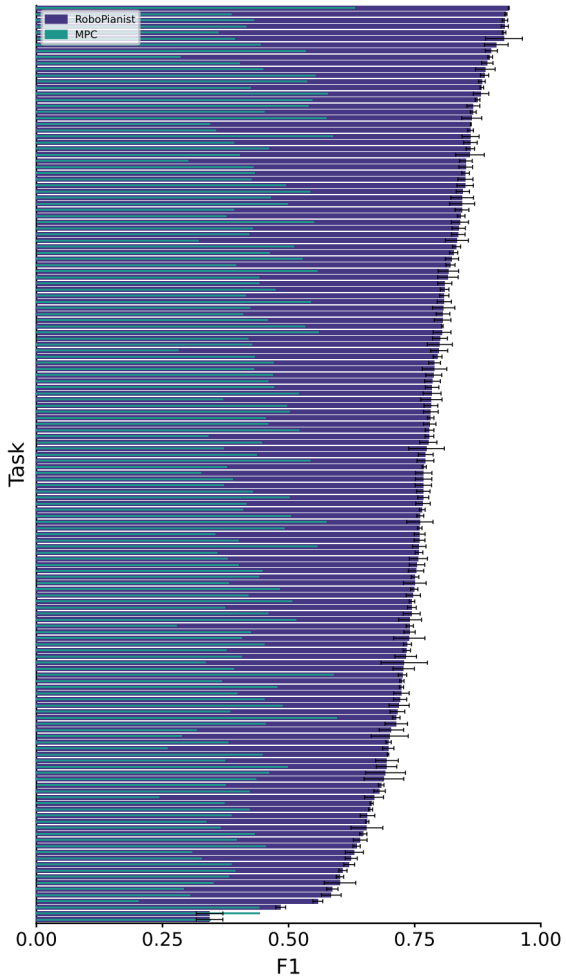

图 6. 原文的 F1 score

4.4 创新点

1. 将原来的代码从 linux/mac 可行，扩展到 windows/linux/mac 全平台可行。

2. 修改了强化学习的过程，将每一个 epoch 都对完整的曲目进行学习，修改为了双层循环，先对完整曲目进行一次学习，然后内部循环 N 次，每次对最低分的 L 帧进行学习，实验表明提高了学习的速度。

3. 修改了 Menagrie Hand 对腕关节的自由度设置，使得机器手能够做两只手非常靠近的那些动作，这样的动作在弹钢琴中会时不时地出现。

5 实验结果分析

5.1 ROBOPIANIST 与 MPC baseline

为了探究本文提出的 ROBOPIANIST 框架是否真的对钢琴演奏任务的提升起到了很大的作用，将它与 Howell 等人使用的 MPC 进行了对比，结果是表现确实超出 MPC 相当的多，原文超出了 83%，我的复现结果，在 32 首歌曲上，表现均有较大幅度的提升。图 6为原文的结果对比，图 7为我的复现对比结果，实验证明，ROBOPIANIST 确实在很大程度上提升了学习的效果。

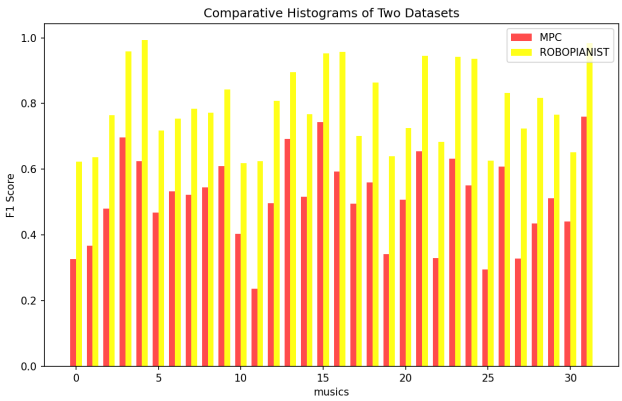

图 7. 我的复现 F1 score

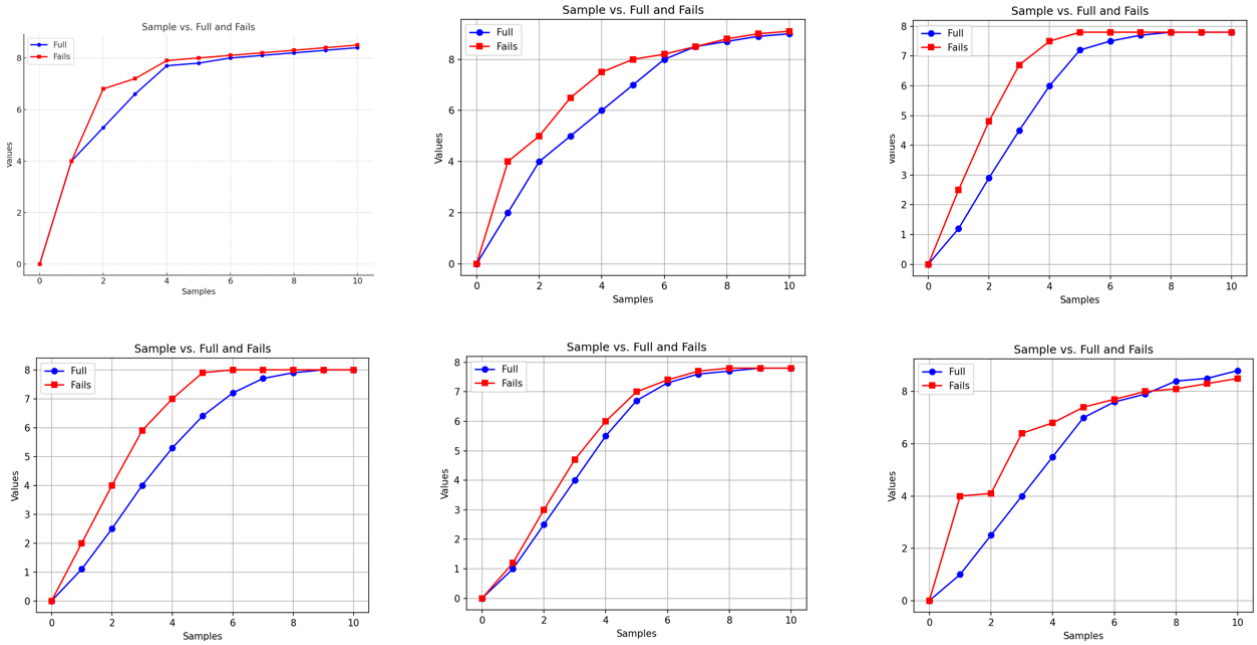

图 8. twincle twincle

5.2 使用 learning from failure 进行学习

这里列出使用 learning from failure 学习 6 首歌曲的效果与不使用 learning from failure 进行学习的效果的学习曲线对比，如图 8。(这六首歌从上到下，从左到右分别是：Twinkle Twinkle、Piano Sonata、French Suite、French Suite、Kreisleriana、Golliwoggs Cakewalk。) 可以看到采用了从失败中学习的策略之后，学习到更好效果的速度比之前更快了，而且最后，两种策略能达到的最佳效果几乎相同。

5.3 修改 Menagerie 的腕关节角度的自由度

如果按照原来的设置，观测到的两只手最靠近的情况，如图 9。

修改腕关节的角度的自由度之后，观测到的最靠近的情况，如图 10。

可以看出，在修改腕关节的设置之后，机器手能够更自如地做出一些手腕靠近，但是却往两边按键盘的动作了，这样的动作对于一个熟练的琴手来说是经常都会出现的。

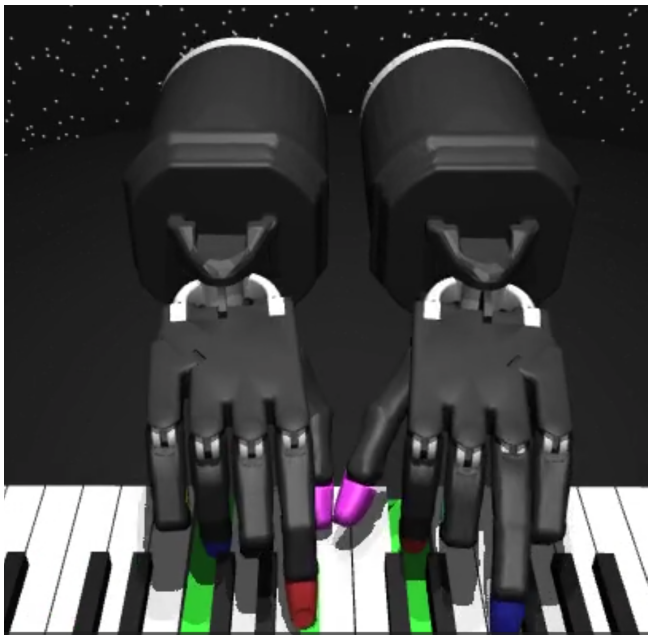

图 9. 原设置

图 10. 修改后

6 总结与展望

本文复现了 ROBOPIANIST: Dexterous Piano Playing with Deep Reinforcement Learning。该研究的主要目的是解决高维控制中实现动作的灵巧性的问题。复现结果表明，该文章提出的一套奖励设计和强化学习的方法是有效的，在很大程度上使得演奏钢琴这样对任意时刻的精确性和整体动作的音乐性有要求的任务对 agent 来说变得可以解决。我在复现本文的过程中，打通了本文章提供的代码无法在 windows 平台上运行的障碍。同时，我在复现的过程中，还创新性的采用了从失败中学习的方法，提高了强化学习的速率。同时我还通过修改手部模型腕关节的自由度设置，使得机器手更好地学习弹钢琴的动作。但是，该文章中提出的采用 Behaivior Cloning 的办法将多个针对单个曲目的专家策略进行蒸馏从而得到一个针对很多歌曲的统一策略的方法效果并不好，而且也不能很好地解决 zero-shot 的泛化能力的问题，我在复现的时候，也没有找到更好的办法来解决这两个问题。后续，我希望能够找到更好的方法来解决学习多首曲目能力变差的问题，和 zero-shot 的泛化能力的问题。

参考文献

[1] Url https://www.shadowrobot.com/dexterous-hand-series/. 2005.

[2] Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, et al. Solving rubik's cube with a robot hand. *arXiv preprint arXiv:1910.07113*, 2019.

[3] OpenAI: Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafal Jozefowicz, Bob McGrew, Jakub Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, et al. Learning dexterous in-hand manipulation. *The International Journal of Robotics Research*, 39(1):3–20, 2020.

[4] Alejandro M Castro, Frank N Permenter, and Xuchen Han. An unconstrained convex formulation of compliant contact. *IEEE Transactions on Robotics*, 39(2):1301–1320, 2022.

[5] Henry J Charlesworth and Giovanni Montana. Solving challenging dexterous manipulation tasks with trajectory optimisation and reinforcement learning. In *International Conference on Machine Learning*, pages 1496–1506. PMLR, 2021.

[6] Tao Chen, Megha Tippur, Siyang Wu, Vikash Kumar, Edward Adelson, and Pulkit Agrawal. Visual dexterity: In-hand dexterous manipulation from depth. In *Icml workshop on new frontiers in learning, control, and dynamical systems*, 2023.

[7] Tao Chen, Jie Xu, and Pulkit Agrawal. A system for general in-hand object re-orientation. In *Conference on Robot Learning*, pages 297–307. PMLR, 2022.

[8] Yuanpei Chen, Tianhao Wu, Shengjie Wang, Xidong Feng, Jiechuan Jiang, Zongqing Lu, Stephen McAleer, Hao Dong, Song-Chun Zhu, and Yaodong Yang. Towards human-

level bimanual dexterous manipulation with reinforcement learning. *Advances in Neural Information Processing Systems*, 35:5150–5163, 2022.

[9] Alejandro Escontrela, Xue Bin Peng, Wenhao Yu, Tingnan Zhang, Atil Iscen, Ken Goldberg, and Pieter Abbeel. Adversarial motion priors make good substitutes for complex reward functions. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 25–32. IEEE, 2022.

[10] Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

[11] Ankur Handa, Arthur Allshire, Viktor Makoviychuk, Aleksei Petrenko, Ritvik Singh, Jingzhou Liu, Denys Makoviichuk, Karl Van Wyk, Alexander Zhurkevich, Balakumar Sundaralingam, et al. Dextreme: Transfer of agile in-hand manipulation from simulation to reality. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5977–5984. IEEE, 2023.

[12] Nicolas Heess, Dhruva Tb, Srinivasan Sriram, Jay Lemmon, Josh Merel, Greg Wayne, Yuval Tassa, Tom Erez, Ziyu Wang, SM Eslami, et al. Emergence of locomotion behaviours in rich environments. *arXiv preprint arXiv:1707.02286*, 2017.

[13] Raymond R Ma and Aaron M Dollar. On dexterity and dexterous manipulation. In *2011 15th International Conference on Advanced Robotics (ICAR)*, pages 1–7. IEEE, 2011.

[14] Josh Merel, Leonard Hasenclever, Alexandre Galashov, Arun Ahuja, Vu Pham, Greg Wayne, Yee Whye Teh, and Nicolas Heess. Neural probabilistic motor primitives for humanoid control. *arXiv preprint arXiv:1811.11711*, 2018.

[15] Josh Merel, Saran Tunyasuvunakool, Arun Ahuja, Yuval Tassa, Leonard Hasenclever, Vu Pham, Tom Erez, Greg Wayne, and Nicolas Heess. Catch & carry: reusable neural controllers for vision-guided whole-body tasks. *ACM Transactions on Graphics (TOG)*, 39(4):39–1, 2020.

[16] Igor Mordatch, Emanuel Todorov, and Zoran Popović. Discovery of complex behaviors through contact-invariant optimization. *ACM Transactions on Graphics (ToG)*, 31(4):1–8, 2012.

[17] Anusha Nagabandi, Kurt Konolige, Sergey Levine, and Vikash Kumar. Deep dynamics models for learning dexterous manipulation. In *Conference on Robot Learning*, pages 1101–1112. PMLR, 2020.

[18] Allison M Okamura, Niels Smaby, and Mark R Cutkosky. An overview of dexterous manipulation. In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, volume 1, pages 255–262. IEEE, 2000.

[19] Tao Pang, HJ Terry Suh, Lujie Yang, and Russ Tedrake. Global planning for contact-rich manipulation via local smoothing of quasi-dynamic contact models. *IEEE Transactions on robotics*, 2023.

[20] Xue Bin Peng, Ze Ma, Pieter Abbeel, Sergey Levine, and Angjoo Kanazawa. Amp: Adversarial motion priors for stylized physics-based character control. *ACM Transactions on Graphics (ToG)*, 40(4):1–20, 2021.

[21] Haozhi Qi, Ashish Kumar, Roberto Calandra, Yi Ma, and Jitendra Malik. In-hand object rotation via rapid motor adaptation. In *Conference on Robot Learning*, pages 1722–1732. PMLR, 2023.

[22] Ilija Radosavovic, Xiaolong Wang, Lerrel Pinto, and Jitendra Malik. State-only imitation learning for dexterous manipulation. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7865–7871. IEEE, 2021.

[23] Aravind Rajeswaran, Vikash Kumar, Abhishek Gupta, Giulia Vezzani, John Schulman, Emanuel Todorov, and Sergey Levine. Learning complex dexterous manipulation with deep reinforcement learning and demonstrations. *arXiv preprint arXiv:1709.10087*, 2017.

[24] Benjamin Scholz. *Playing piano with a shadow dexterous hand*. PhD thesis, Universität Hamburg, 2019.

[25] Archit Sharma, Shixiang Gu, Sergey Levine, Vikash Kumar, and Karol Hausman. Dynamics-aware unsupervised discovery of skills. *arXiv preprint arXiv:1907.01657*, 2019.

[26] Christian Smith, Yiannis Karayiannidis, Lazaros Nalpantidis, Xavi Gratal, Peng Qi, Dimos V Dimarogonas, and Danica Kragic. Dual arm manipulation—a survey. *Robotics and Autonomous systems*, 60(10):1340–1353, 2012.

[27] Laura Smith, Ilya Kostrikov, and Sergey Levine. A walk in the park: Learning to walk in 20 minutes with model-free reinforcement learning. *arXiv preprint arXiv:2208.07860*, 2022.

[28] Balakumar Sundaralingam and Tucker Hermans. Relaxed-rigidity constraints: kinematic trajectory optimization and collision avoidance for in-grasp manipulation. *Autonomous Robots*, 43:469–483, 2019.

[29] Saran Tunyasuvunakool, Alistair Muldal, Yotam Doron, Siqi Liu, Steven Bohez, Josh Merel, Tom Erez, Timothy Lillicrap, Nicolas Heess, and Yuval Tassa. dm_control: Software and tasks for continuous control. *Software Impacts*, 6:100022, 2020.

[30] Huazhe Xu, Yuping Luo, Shaoxiong Wang, Trevor Darrell, and Roberto Calandra. Towards learning to play piano with dexterous hands and touch. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10410–10416. IEEE, 2022.