Dokumentace ročníkového projektu

Vývoj řízení simulovaných robotů ve 3D prostředí

Marek Bečvář
MFF UK 2022

Obsah

I Popis a cíl projektu

Projekt je zaměřen na využití genetických algoritmů pro vývoj řízení simulovaných robotů ve fyzikálním prostředí. Řízení se má vyvíjet směrem k předem specifikovanému cíli. Tímto cílem může být mimo jiné například uražená vzdálenost nebo maximální rychlost pohybu. Cílem projektu je seznámení se s různými možnostmi zvoleného fyzikálního prostředí, vývoj základního genetického algoritmu a jeho aplikace na sadu výchozích a vlastních simulovaných robotů a vytvoření další sady aplikací umožňující statistické zpracování výsledků.

II Dostupné technologie

i. Fyzikální simulátory

MuJoCo (*Multi-Joint Dynamics with Contact*) je zdarma a open source robustní fyzikální engine pro vývoj v oblasti robotiky, biomechaniky a dalších. MuJoCo se dále často využívá pro testování a porovnávání různých metod navrhování robotických systémů jako jsou třeba evoluční algoritmy nebo reinforcement learning [1].

MuJoCo umožňuje velký nárůst v rychlosti běhu simulace za pomoci plné podpory paralelizace na všech dostupných jádrech a stabilitě simulace i při využití větších simulačních časových kroků [2]. Zároveň nabízí jednoduchý styl, jakým si může uživatel upravit všechny detaily simulace i robotů samotných pomocí C++ API nebo jednoduchých XML konfiguračních souborů.

Webots Webots je open source multiplatformní robotický simulátor s využitím v průmyslu, výuce i výzkumu [3]. Webots umožňuje programování a testování jak jednoduchých virtuálních robotů, tak následné aplikace softwaru na reálné roboty, obojí využívající programovacího jazyka C a stejných *Khepera* API[4].

Prostředí nabízí využití řady předpřipravených modelů robotů všech druhů a dále umožňuje import vlastní robotů z 3D modelovacích softwarů v CAD formátu.

Player/Stage Player/Stage je výsledkem projektu *The Player Project* [5] vytvořeným za účelem usnadnit vývoj v oblasti robotiky a senzorů. Player/Stage je spojení dvou projektů.

Player je síťový jazykově a na platformě nezávislý server pro ovládání robota s jednoducým přístupem k senzorům a motorům robota skrz IP síť.

Klientský program může být vytvořený a spuštěný na libovolném PC s připojením k síti robota a v libovolném jazyce podporujícím TCP sokety.

Stage je rychlý a rozšířitelný 2D simulátor prostředí s objekty a roboty s jejich senzory, kam roboti a jejich ovladače mohou být načítány za běhu simulace.

Player/Stage Často jsou tyto dva programy využívané dohromady tak, že uživatel vyvine populaci robotů (ovladačů, senzorů) a poskytne ji jako klienty pro Player server.

Gazebo Gazebo je sada open source knihoven pro vývoj, výzkum a aplikaci robotů. Umožňuje simulaci dynamického 3D prostředí s více agenty, generování reálných dat ze simulovaných senzorů a fyzikálně korektní interakce robotů s prostředím [6]. Pro simulace umožňuje výběr z více fyzikálních enginů.

Uživatel pracuje a nastavuje prostředí v interním grafickém prostředí s určitou možností spouštět simulace i bez GUI.

Gazebo bylo součástí projektu *Player Project* od roku 2004, ale od roku 2012 je nezávislým projektem *Open Robotics* [7] [5].

ii. Evoluční algoritmy

DEAP DEAP (*Distributed Evolutionary Algorithms in Python*) je Python framework pro tvorbu evolučních algoritmů, který se snaží jejich tvorbu zjednodušit pomocí přímočarého postupu (podobného pseudokódu), který je jednoduchý na porozumění. [8]

Framework je tvořený ze dvou hlavních struktur *creator*, který pomáhá s vytvářením genetických informací z jedinců z libovolných datových struktur a *toolbox*, který je seznamem nástrojů (genetických operátorů), které mohou být použité při tvorbě algoritmu.

Inspyred Inspyred poskytuje implementaci většiny z nejpoužívanějších evolučních algoritmů a dalších přírodou inspirovaných algoritmů v jazyce Python. [9]

Knihovna již přichází s funkčním řešením, ve formě jednotlivých komponentů (python funkcí), které ale uživatel může sám přepsat, nebo nahradit za jiné již vytvořené funkce.

III Použité technologie

- MuJoCo
- Vlastní evoluční algoritmus
- OpenAI - Gym (Python API pro vývoj AI v různých prostředích)
- Python (Programovací jazyk)
- XML

OpenAI - Gym OpenAI je firma zaměřená na výzkum, vývoj a praktické využití umělé inteligence. **Gym** je open source Python API firmy OpenAI. Je to platforma pro vývoj převážně reinforcement learning metod [10]. Umožňuje využít řadu prostředí, ve kterých uživatelé mohou jednoduše spouštět a testovat své agenty [10]. Tato prostředí mohou být
ku příkladu Atari hry, textové hry, jednoduché 2D i plně fyzikálně simulované 3D prostředí (s fyzikálně enginem **MuJoCo**).

Gym nabízí jednoduchý přístup do všech těchto prostředí kde vstupy (akce agenta v prostředí) i výstupu (stav prostředí, pozorování agenta) jsou standardizované napříč všemi prostředími. Navíc open source vlastnost
tohoto API umožňuje vlastní doprogramování pokročilých pomocných nástrojů pro vývoj a práci s prostředím.

I když je Gym primárně vytvořené pro vývoj reinforcement learning agentů, je velmi jednoduché použít namísto toho agenta, který je v našem případě vyvíjen pomocí evolučních algoritmů.

i. Odůvodnění vybraných technologií

Fyzikální simulátor - MuJoCo MuJoCo bylo oproti ostatním simulátorům zvoleno, kvůli jednoduchosti konfigurace celého prostředí.
Dále spojení s prostředím skrz open source API od OpenAI umožňuje přímočaré upravování všech částí prostředí, což velmi pomáhá při vývoji vlastních evolučních agentů. Dále, jelikož evoluční algoritmy jsou zdlouhavým procesem, rychlost a možná paralelizace běhu prostředí byla při výběru
pozitivní vlastností. S ohledem na možné budoucí experimenty zároveň bylo žádoucí mít možnost jednoduché úpravy všech aspektů simulovaných robotů a jejich prostředí. Zde XML konfigurační soubory, které MuJoCo používá, pomocí kterých může uživatel postavit celého robota a zároveň skrz ně
nastavit kompletní vlastnosti prostředí byly přínosem pro rychlejší experimentování a vývoj.

Využití vlastního evolučního algoritmu Pro jednoduchost implementace a snížení nároků na znalosti mnohdy velmi složitých externích knihoven pro evoluční algoritmy, které krom zkrácení zápisu často nepřináší více dalších pozitivních vlastností, bylo rozhodnuto o využití vlastní jednoduché implementace všech základních operátorů evolučních algoritmů, které je možné poté v implementaci vlastních experimentálních agentů jednoduše aplikovat a používat bez potřeby znalostí externích knihoven.

Programovací jazyk - Python Pro implementaci všeho je využitý programovací jazyk Python. Důvodem byla potřeba nějakého jednoduchého, nejlépe interpretovaného jazyka, vhodného pro rychlé experimentování a dobré pro předvádění finálních výsledků.

Odkazy

- Oficiální Gym library docs
- Gym Github
- Getting started with OpenAI Gym
- Medium článek - Exploring OpenAI Gym
- Python.org
- Python - Wikipedia
- XML Introduction - Mozilla Developer
- XML - Wikipedia

IV Popis softwarového díla

Průběh experimetnu Nejprve je potřeba pomocí XML souboru zvolit určitého definovaného robota pro náš experiment. Dále mu přiřadíme vytvořeného agenta, který ovlivňuje jak jsou do genetického algoritmu kódovány vstupy z prostředí libovolného robota. Tento agent má předem přiřazené genetické operátory, které ale uživatel může dle potřeb experimentu zaměňovat za jiné, buď předem definované, nebo zcela vlastní vytvořené operátory ve tvaru odpovídajících Python funkcí. Dále máme možnost takto připravenému prostředí nakonfigurovat vlastní fitness funkci a nastavit

chtěnou dobu trvání experimentu (počet generací, které má algoritmus projít). Rozběhlé prostředí jsme pak schopni pozorovat pomocí aktivních statistik, které nám dávají představu o hodnotách z vývoje v prostředí. Pokud uživatel chce mít větší přehled o vývoji v prostředí, má možnost určité generace (jedince z generace) interaktivně sledovat skrz kamery v simulovaném prostředí, ve kterém experiment probýhá. Jelikož experimenty pro dosažení nějakých přínosných výsledků je potřeba pro kontrolu opakovat, umožňuje program opakované spouštění experimentů s předem zvolenými parametry (úprava parametrů možna jak pro prostředí, tak pro samotného agenta a jeho genetické operátory). Všechny tyto výsledky jsou pak podle zvolených parametrů ukládány, umožňující následné externí zpracování, popřípadě využití jednoduchých statistických pomůcek (vytvořených v rámci tohoto projektu) pro rozbor dat z výsledků běhů algoritmů v grafech. Z každého běhu je zároveň ukládán nejlepší jedinec, pro umožnění vizuální kontroly a rozboru výsledného jedince finální generace.

References

[1] Tim Salimans, Jonathan Ho, Xi Chen, Szymon Sidor, and Ilya Sutskever. Evolution strategies as a scalable alternative to reinforcement learning. *arXiv preprint arXiv:1703.03864*, 2017.

[2] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 5026–5033. IEEE, 2012.

[3] Webots. http://www.cyberbotics.com. Open-source Mobile Robot Simulation Software.

[4] Olivier Michel. Webots: Symbiosis between virtual and real mobile robots. In *International Conference on Virtual Worlds*, pages 254–263. Springer, 1998.

[5] The player project. http://playerstage.sourceforge.net/.

[6] N. Koenig and A. Howard. Design and use paradigms for gazebo, an open-source multi-robot simulator. In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, volume 3, pages 2149–2154 vol.3, 2004.

[7] Gazebo. https://gazebosim.org/.

[8] Félix-Antoine Fortin, François-Michel De Rainville, Marc-André Gardner Gardner, Marc Parizeau, and Christian Gagné. Deap: Evolutionary algorithms made easy. *The Journal of Machine Learning Research*, 13(1):2171–2175, 2012.

[9] Alberto Tonda. Inspyred: Bio-inspired algorithms in python. *Genetic Programming and Evolvable Machines*, 21(1):269–272, 2020.

[10] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.