

GYMNÁZIUM JANA KEPLERA
Praha 6, Parlérova 2
obor vzdělání Gymnázium 79-41-K/81

Maturitní práce z informatiky

Vývoj chůze pomocí neuroevolučních algoritmů

Jan Bouček
vedoucí práce: Tomáš Obdržálek
Praha 2017

Prohlašuji, že jsem jediným autorem této maturitní práce a všechny citace, použité literatura a další zdroje jsou v práci uvedené. Tímto dle zákona 121/2000 Sb. (tzv. Autorský zákon) ve znění pozdějších předpisů uděluji bezúplatně škole Gymnázium Jana Keplera, Praha 6, Parlérova 2 oprávnění k výkonu práva na rozmnožování díla (§ 13) a práva na sdělování díla veřejnosti (§ 18) na dobu časově neomezenou a bez omezení územního rozsahu.

V Praze dne 24. 3. 2017

Jan Bouček

Obsah

1	Úvod	4
2	Teorie	4
2.1	Evoluční algoritmy	4
2.2	Neuroevoluce	4
2.2.1	Neuronové sítě	4
2.2.2	NEAT	5
2.3	CPPN-NEAT	5
2.4	HyperNEAT	6
2.5	Rozložení sítě	6
2.6	simulace	7
3	Výsledky	7
4	Uživatelská příručka	7
5	Závěr	7
6	Diskuze?	7

Anotace

Jeden z problémů moderní robotiky je chůze, čehož se tato práce narozdíl od konvenčních metod snaží dosáhnout pomocí strojového učení, konkrétně neuroevolučního algoritmu s nepřímým kódováním. Ten za pomoci techniky HyperNEAT generuje simulaci primitivního mozku, který chůzi robota ovládá.

Annotation

One of the problems in modern robotics is robotic walking, which in this paper, unlike conventional methods, is achieved by machine learning, specifically using a neuroevolutionary algorithm with indirect encoding. The algorithm uses the HyperNEAT technique to generate a simulation of a primitive brain, which controls the gait of the robot.

Vývoj chůze pomocí neuroevolučních algoritmů

Jan Bouček

1 Úvod

Jeden z cílů dnešní robotiky je konstruovat roboty, kteří jsou všestranně využitelní díky své mobilitě. Toho lze dosáhnout různými způsoby, pro svou jednoduchost se často používá jízda, která ale neumožňuje spolehlivé zdolávání členitého terénu, nebo pohyb přímo ve strukturách stavěných člověkem, zejména pak zdolávání schodů.

Tato práce se inspirovuje čtyřnohými kráčivými roboty firmy Boston Dynamics[1], kteří dosahují stabilní chůze exaktními metodami.[2] Podobné chůze by mělo být možné dosáhnout i pomocí organičtějších metod. Neuroevoluční algoritmy dokáží nabídnout přirozené řešení i na velmi komplexní problémy, či dokonce znovuobjevit řešení, které v přírodě už existuje, proto je možné generovat čtyřnohý pohyb i tímto způsobem.[3]

Cílem projektu pak je dosáhnout podobnou metodou stabilní, vzpřímené a repetitivní kupředné chůze na simulaci čtyřnohého robota.

2 Teorie

Chůzi lze definovat jakožto sérii stavů kráčivého tělesa, kde každý stav lze zjednodušit jako soubor informací popisující jednotlivé pohyblivé prvky robota. V rámci tohoto projektu budeme pracovat s dvojrozměrnou simulací čtyřnohého robota, jehož každá noha je ovládána dvěma klouby. Stav robota tak lze vystihnout jako osmirozměrný vektor \vec{s} . Hledáme pak ovladač robota, který pro každý stav malezne osmirozměrný vektor $\vec{\omega}$, který každému kloubu určuje úhlovou rychlost.

2.1 Evoluční algoritmy

Evoluční algoritmy jsou metodou, která dokáže mnohásobnou selekcí, mutací a kombinací různých řešení postupně vyvíjet optimálnější. K tomu potřebuje funkci, která dokáže každé řešení ohodnotit, tzv. *fitness funkci*. V našem případě hodnotíme maximální vzdálenost robota od počátku směrem do kladných hodnot x , a vzhledem k tomu, že cílem je chůze vzpřímená, tak u každého řešení, ve kterém některý článek robota klesne svou výškou těžiště pod experimentálně zjištěnou hodnotu Y_{min} , nastavíme výslednou *fitness* ovladače na 0.

Podobně jako v přírodě, evoluční algoritmus si udržuje *populaci* řešení. V každém kroku ohodnotí výše zmíněnou funkcí každého *živočicha*, podle své úspěšnosti jsou pak rozmnoženi (pohlavně, či nepohlavně) a zmutováni. Tento proces je podrobněji popsán v následující sekci.

2.2 Neuroevoluce

Neuroevoluční algoritmy jsou specifickou kategorií, která se zabývá trénováním umělých neuronových sítí, anglicky *Artificial Neural Network* (ANN), které jsou abstraktním přiblížením dějů v mozku, měly by být tedy schopné sloužit i jako ovladač našeho robota. Jejich struktura je popsána níže.

2.2.1 Neuronové sítě

Základní stavební jednotkou neuronové sítě je *neuron*, ten je výpočetní jednotkou, která přijímá data ve formě spojů (z biologického hlediska *dendrity*) s dalšími neurony. Pro každý z nich má nastavenou *váhu* (anglicky *weight*), která slouží jednoduše jako koeficient, který ovlivňuje „důležitost“ konkrétního spoje.

Výstupem, který pak mohou použít další neurony, je výsledek *aktivační funkce* lineární kombinace hodnot napojených neuronů a příslušných vah[4]:

$$y_k = \phi(\sum_{j=0}^m w_{kj} \cdot x_j),$$

kde w je váha hodnoty neuronu x a ϕ aktivační funkce.

Neurony pak lze tímto způsobem zapojovat do komplexních sítí, kde hodnotu některých určujeme manuálně – považujeme je za *vstupy* sítě a u některých odečítáme hodnoty, čímž fungují jako *výstupy* sítě.

2.2.2 NEAT

Neuroevolution of augmenting topologies[5] (NEAT) je metodou, která definuje jeden ze způsobů, kterými lze vyvíjet neuronové sítě pomocí evolučního algoritmu. Revoluční ideou tohoto systému je způsob, kterým zapisuje neuronové sítě, jakožto *živočicha*.

Každou neuronovou síť popisuje jako ohodnocený graf pomocí dvou seznamů *genů* – seznam vrcholů a seznam hran. Každému vrcholu a hraně přisuzuje číselné identifikátory, pomocí kterých je dokáže přehledně propojovat a udržovat přehled o tom, kteří jedinci jsou „geneticky“ podobní.

Stanley a Miikkulainen díky tomu zavádí i proces *speciace*, který ještě před rozmnožováním roztrídí jedince do různých *druhů* podle příbuznosti tak, aby se spolu křížily jen sítě s menšími rozdíly. Každý druh je pak ohodnocen svojí průměrnou hodnotou fitness, pomocí které se určí počet potomků v další generaci daného druhu.

Při procesu rozmnožování se z každého druhu vybere *silnější* část, ze které se vytvoří požadovaný počet potomků, z nichž každý může vzniknout křížením – většinu genů zdědí po silnějším rodiči, ale část od slabšího, nebo bez křížení, kdy se pouze jedinec zkopíruje do další generace.

Pak na všech potomcích proběhne mutace, přičemž je náhodně rozhodnuto, které z druhů mutací na něm proběhnou, Stanley a Miikkulainen popisují hned několik:

- přidání nové hrany do sítě
- rozdělení hrany na dvě hrany s novým neuronem uprostřed
- změna všech vah o malou hodnotu, nebo na náhodnou hodnotu

V tomto projektu je použito ještě několik dalších mutačních operátorů:

- aktivace/deaktivace jedné hrany
- změna jedné váhy na náhodnou hodnotu
- změna jedné váhy o až $\pm 5\%$

Díky této mírné úpravě dokáže algoritmus ladit neuronovou síť na trochu detailnějším měřítku.

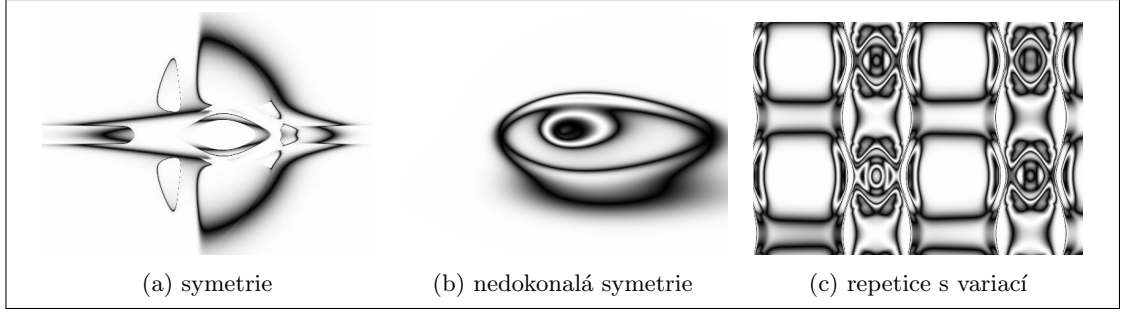
Po zmutování jsou všichni potomci prohlášeni za současnou generaci a algoritmus pokračuje znovu hodnocením jedinců.

2.3 CPPN-NEAT

Tím, že v NEAT algoritmu umožníme každému neuronu využívat jinou aktivační funkci, můžeme dosáhnout sítí, které jsou dobře uzpůsobené ke generování fyzické geometrie živočichů. Metoda CPPN-NEAT[6] využívá různých vlastností funkcí – symetrie, repetice apod. tak, že průchodem celou sítí jsou výsledná data nakombinována pomocí komplexní složené funkce, která si uchovává všechny tyto vlastnosti.

Pokud například vytvoříme CPPN síť s dvěma vstupy – souřadnice x a y s jedním výstupem, získáme dvojrozměrné útvary (viz obr. 1), které mají velmi blízko k fyzickému rozložení v reálných živočíchách, např. dokáže nakombinovat repetici s variací a generovat útvary podobné prstům u ruky nebo pomocí symetrie a variace útvary podobné lidskému oku (viz obr. 1b)

V samotném algoritmu stačí jen při tvoření nových neuronů určit náhodnou aktivační funkci a přidat mutační operátor, který změní funkci u náhodného neuronu.



Obrázek 1: tvary vygenerované pomocí CPPN-NEAT, převzato [7]

2.4 HyperNEAT

Technika HyperNEAT [7] využívá CPPN sítě ke generování čtyřrozměrného prostoru, který ve výsledku slouží jako definice vah v další neuronové síti.

To znamená, že generujeme CPPN síť, která má čtyři vstupy – x_1, y_1, x_2, y_2 . Výstup nám pak určuje váhu spoje z neuronu na fyzických souřadnicích $[x_1; y_1]$ do neuronu na souřadnicích $[x_2; y_2]$. Stačí nám pak vymyslet fyzickou strukturu sítě, kterou využijeme této techniky.

V tomto projektu je použito rozložení zvané „state-space sandwich“ [7] o třech vrstvách podobně jako [3], kde výsledná síť je rozvrstvená do trojrozměrného prostoru a z každého neuronu vede spoj do každého neuronu v další vrstvě. CPPN síť má pak dva výstupy, kde první určuje váhu mezi danými souřadnicemi z první vrstvy do druhé a druhý výstup určuje váhy mezi druhou a třetí vrstvou.

2.5 Rozložení sítě

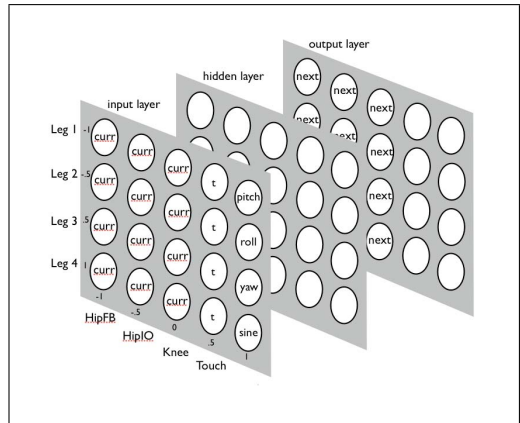
Narozdíl od klasických neuronových sítí jsou si sítě generované HyperNEATEm vědomé geometrických souvislostí [7], proto je naprosto zásadní rozmístění vstupů a výstupů ve výsledné síti.

[3] umísťuje do každého řádku v substrátu jednu nohu a do posledního sloupce přidává neuspořádané zbylé informace (viz obr. 2).

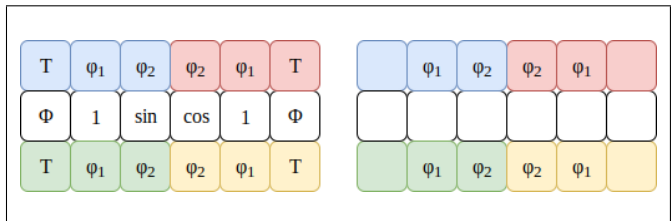
Tento postup sice dává geometricky najevo související buňky – podobné informace jsou ve stejných sloupcích, ale opomíjí rozložení nohou v prostoru, proto se v tomto projektu používá struktura, která za pomoci symetrie dává najevo souvislosti typu přední/zadní pár nohou a levá/pravá noha. Pro nejlepší výsledky je v rozložení nohou středová souměrnost a zbylé informace jsou v prostřední řadě mezi nohami. Každá noha je popsána pomocí dvou úhlů kloubů a dotykového senzoru na spodním článku nohy, jehož hodnota je nastavena na -1 bez doteku a 1 s dotekem, pro hladkost pohybu mají tyto hodnoty lineární přechod o délce 25 snímků.

Jako přídatné proměnné jsou v tomto rozložení $\sin(t/T)$, $\cos(t/T)$, kde T je experimentálně určená konstanta – 40, dále úhel těla robota Φ a 1 jakožto lineární konstanta.

Z výstupní vrstvy se využívají pouze neurony v pozicích úhlů nohou, které jsou převedeny na úhlovou rychlost podobně jako v [3]: $\omega = 4,5 \cdot (\phi_{new} - \phi_{cur})$



Obrázek 2: rozložení sítě v [3], převzato [3]



Obrázek 3: symetrické rozložení sítě, barvy označují jednotlivé nohy, vlevo vstupní vrstva, vpravo výstupní vrstva, prostřední skrytá vrstva nevyobrazena

2.6 simulace

K simulaci je použita knihovna JBox2D[8], robot se skládá z hlavního těla, na které jsou napojeny nohy. Každá z nich je napojena pomocí rotujícího *RevoluteJointu* a stejným způsobem je napojen i spodní segment v koleni.

Měření času probíhá 3000 snímků a dalších 500 snímků se čeká, jestli robot po konci měření upadne na zem. Tím se zabraňuje strategiím, kde místo snahy o chůzi robot pouze skočí směrem kupředu.

3 Výsledky

4 Uživatelská příručka

5 Závěr

Reference

- [1] Introducing Spot. In: *Youtube* [online]. 9.2.2015 [cit. 2017-03-20]. Dostupné z: <https://www.youtube.com/watch?v=M8YjvHYbZ9w>. Kanál uživatele Boston Dynamics
- [2] RAIBERT, Marc, Kevin BLANKESPOOR, Gabriel NELSON a Rob PLAYTER. *BigDog, the Rough-Terrain Quadruped Robot* [online]. [cit. 2017-03-20]. DOI: 10.3182/20080706-5-KR-1001.01833. ISBN 10.3182/20080706-5-KR-1001.01833. Dostupné z: <http://linkinghub.elsevier.com/retrieve/pii/S1474667016407020>
- [3] CLUNE, Jeff, Benjamin E. BECKMANN, Charles OFRIA a Robert T. PENNOCK. Evolving coordinated quadruped gaits with the HyperNEAT generative encoding. In: *2009 IEEE Congress on Evolutionary Computation*. IEEE, 2009, s. 2764-2771. DOI: 10.1109/CEC.2009.4983289. ISBN 978-1-4244-2958-5. Dostupné také z: <http://ieeexplore.ieee.org/document/4983289/>
- [4] Artificial neuron. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001-2017 [cit. 2017-03-21]. Dostupné z: https://en.wikipedia.org/wiki/Artificial_neuron
- [5] STANLEY, Kenneth O. a Risto MIKKULAINEN. Evolving Neural Networks through Augmenting Topologies. *Evolutionary Computation* [online]. 2002, **10**(2), 99-127 [cit. 2017-03-21]. DOI: 10.1162/106365602320169811. ISSN 1063-6560. Dostupné z: <http://www.mitpressjournals.org/doi/10.1162/106365602320169811>
- [6] STANLEY, Kenneth O. Compositional pattern producing networks: A novel abstraction of development. *Genetic Programming and Evolvable Machines* [online]. 2007-6-6, **8**(2), 131-162 [cit. 2017-03-21]. DOI: 10.1007/s10710-007-9028-8. ISSN 1389-2576. Dostupné z: <http://link.springer.com/10.1007/s10710-007-9028-8>
- [7] STANLEY, Kenneth O., David B. D'AMBROSIO a Jason GAUCI. A Hypercube-Based Encoding for Evolving Large-Scale Neural Networks. *Artificial Life* [online]. 2009, **15**(2), 185-212 [cit. 2017-03-21]. DOI: 10.1162/artl.2009.15.2.15202. ISSN 1064-5462. Dostupné z: <http://www.mitpressjournals.org/doi/10.1162/artl.2009.15.2.15202>
- [8] *JBox2D* [online]. Daniel Murphy, 2014 [cit. 2017-03-22]. Dostupné z: <http://www.jbox2d.org/>