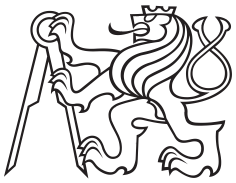
Bakalářská práce



České
vysoké
učení technické
v Praze

F3

Fakulta elektrotechnická

Reprezentace motorických akcí pro imitační učení v robotice

Kateřina Kubecová

Školitel: Mgr. Karla Štěpánová, Ph.D
Školitel–specialista: Mgr. Gabriela Šejnová
Květen 2022

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Kubecová** Jméno: **Kateřina** Osobní číslo: **492171**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra počítačů**

Studijní program: **Otevřená informatika**

Specializace: **Software**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Reprezentace motorických akcí pro imitační učení v robotice

Název bakalářské práce anglicky:

Representation of motoric actions for imitation learning in robotics

Pokyny pro vypracování:

Ke správnému naučení pohybu na základě demonstrace a jeho zobecnění do nového prostředí je důležitý výběr vhodné reprezentace získaných senzorických vstupů a pečlivý výběr proměnných, které by měly danou akci (pohyb) reprezentovat.
1) Vytvořte přehled současně používaných metod pro reprezentování akcí.
2) V simulátoru CoppeliaSim vytvořte prostředí, ve kterém je možné generovat jednotlivé parametrizované akce a sestavte set objektů s různými vlastnostmi (např. váha, tvar, velikost, povrchová textura), na kterých budete akce demonstrovat.
3) Nahrajte dataset s jednoduchými motorickými akcemi ("posuň objekt", "zvedni objekt", "otoč objekt", atd.) demonstrovanými na vybraném setu objektů tak, aby obsahoval všechny informace potřebné pro odlišení jednotlivých typů akcí (např.: pozice objektu, síla aplikovaná na objekt).
4) Na vámi zvolené metodě strojového učení ukažte, které příznaky z naměřeného datasetu jsou podstatné pro odlišení a naučení daných akcí.

Seznam doporučené literatury:

- Fang, Bin, et al. "Survey of imitation learning for robotic manipulation." International Journal of Intelligent Robotics and Applications 3.4 (2019): 362-369.
- Herath, Samitha, Mehrtash Harandi, and Fatih Porikli. "Going deeper into action recognition: A survey." Image and vision computing 60 (2017): 4-21.
- Kong, Yu, and Yun Fu. "Human action recognition and prediction: A survey." arXiv preprint arXiv:1806.11230 (2018).
- Krüger, V., Kragic, D., Ude, A., & Geib, C. (2007). The meaning of action: A review on action recognition and mapping. Advanced robotics, 21(13), 1473-1501.
- Beddiar, D. R., Nini, B., Sabokrou, M., & Hadid, A. (2020). Vision-based human activity recognition: a survey. Multimedia Tools and Applications, 79(41), 30509-30555.

Jméno a pracoviště vedoucí(ho) bakalářské práce:

Mgr. Karla Štěpánová, Ph.D. robotické vnímání CIIRC

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce: **02.02.2022** Termín odevzdání bakalářské práce: **20.05.2022**

Platnost zadání bakalářské práce: **30.09.2023**

Mgr. Karla Štěpánová, Ph.D. podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	prof. Mgr. Petr Páta, Ph.D. podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Studentka bere na vědomí, že je povinna vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

. _____
Datum převzetí zadání

_____
Podpis studentky

Poděkování

Zde bych chtěla poděkovat paní Mgr. Karle Štěpánové Ph.D. za vedení této práce a Mgr. Gabriele Šejnové za cenné rady. Dále bych chtěla poděkovat svému strýci a tětě za podporu během celého studia.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracovala samostatně a že jsem uvedla veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze, . května 2022

Abstrakt

Tato práce se zabývá manipulací s předměty pomocí robotické ruky. Zaměřujeme se na to, jak důležitý je výběr proměnných reprezentujících danou akci (pohyb) pro naučení a zobecnění dané akce do nového prostředí v případě, že využíváme pro naučení akce generativních modelů. V první části práce připravujeme generátor datasetů pro sadu motorických akcí (posuň, zvedni a strč) v robotickém simulátoru CoppeliaSim s využitím robotického sedmiosého manipulátoru. Generátor umožňuje variovat jak způsob provedení akce (např. rychlost pohybu), tak i parametry manipulovaných objektů (např. velikost, váhu, tření či tvar) a výchozí parametry scény (např. úvodní pozice objektu a manipulátoru). V druhé části práce upravujeme generativní model ACTOR tak, aby ho bylo možné podmínit nejen typem akce, ale i parametry manipulovaného tělesa. Na základě připravených evaluačních kritérií hodnotíme kvalitu natrénovaných modelů pro různé typy datasetů (velikost datasetu, tělesa různé velikosti, tělesa různé velikosti a tvaru, atp.).

Klíčová slova: motorické akce, variační autoenkodér, CoppeliaSim

Školitel: Mgr. Karla Štěpánová, Ph.D

Abstract

This work deals with the manipulation of objects using robotic arm. We focus on how important it is to select variables representing a given action (movement) to learn and generalize the action to a new environment if we use generative models to learn the action. In the first part of the work we are preparing a dataset generator for a set of motor actions (poke, lift and push) in the robotic simulator CoppeliaSim using a robotic 7-axis manipulator. The generator allows you to vary both the way the action is performed (eg. movement speed) and the parameters of the manipulated objects (eg. size, weight, friction or shape) and the default parameters of the scene (eg. the initial position of the object and the manipulator). In the second part of the work, we modify the generative ACTOR model so that it is possible to condition not only the type of action, but also the parameters of the manipulated object. Based on the prepared evaluators, we evaluate the quality of trained models for different types of datasets (dataset size, objects of different sizes, objects of different sizes and shapes, etc.).

Keywords: motorical actions, variational autoencoder, CoppeliaSim

Title translation: Representation of motoric actions for imitation learning in robotics

Obsah

Obrázky

Tabulky

Kapitola 1
Úvod

Na rozdíl od počítače mají roboti možnost fyzicky interagovat s prostředím. Někteří roboti se v něm mohou pohybovat, jiní naopak manipulují s předměty. Manipulace s předměty se využívá především v průmyslu, kde roboti opracovávají a přemisťují výrobky.

Aby byli ve střetu s reálným světem efektivní, je třeba, aby byly podmínky pro splnění úkolu stále stejné, nebo aby se robot uměl změnám v prostředí přizpůsobovat.

Uvažujme například robota, jež má umístit plastový kelímek do koše. Kelímky budou naskládané v pravidelných vzdálenostech na jedoucím páse, jeden jako druhý. Pak stačí, když bude robot vykonávat přesně stejnou sekvenci pohybů stále dokola.

Situace se změní, pokud bude robot sbírat kelímky ze stolu. Každý kelímek bude jinde, některé budou těžší, protože v nich zůstane zbytek vody, jiné budou pomačkané. Sekvence pohybů pro splnění úkolu nad jednotlivými kelímky budou sice rámcově podobné, ale rozhodně už ne stejné.

V současné robotice stále převládají předprogramované pohyby. Ve výrobě to znamená, že se i při malé změně výrobku musí program robota upravovat. Úkolem moderní automatizace je, aby se robot vždy přizpůsobil aktuálnímu výrobku, aniž by se změnila výsledná činnost.

Právě o jedné z cest, jak přizpůsobit robotovi pohyby konkrétním předmětům, pojednává tato práce. Konkrétně se zabývá úkoly *posuň* těleso, *strč* do tělesa a *zvedni* těleso. Tělesa mění počáteční polohu, tvar, hmotnost a velikost.

Zabýváme se metodami učení těchto akcí na základě trénovacích příkladů jednotlivých typů akcí. Zaměříme se zejména na vhodné způsoby reprezentací těchto akcí pro efektivní učení pomocí neuronových sítí.

1.1 Motivace

K tématu pohybu a manipulace s předměty mne motivovala moje záliba v ručních pracech. Na první pohled se zdá, že člověk svým rukoum vládne dokonale. Když se ale někdo učí háčkovat nebo plést, nestačí vědět, co provléct kterým okem. Pro pěknou, rychle upletenou řadu jsou zapotřebí

hodiny tréninku. Naučí-li se někdo háčkovat na vlně, měl by to automaticky umět i na bavlnce. Je to totéž, jen v menším. Ani to však nepůjde hned.

Proč i tak výkonná věc, jakou je lidský mozek, potřebuje tutéž věc mnohokrát zopakovat? Které informace jsou pro učení důležité? Co všechno je potřeba pro to, aby bylo možné podobně učit robota?

Odpovědi na tyto a podobné otázky považuji za důležité pro to, aby člověk své činnosti opravdu pochopil a ovládnul.

1.2 Cíle

Cílem této práce je naučit robotickou ruku manipulovat s předměty. Předměty budou měnit svoji polohu a vlastnosti, čemuž se má ruka přizpůsobovat. Pokusy budou vytvářeny pomocí generativního modelu, který je nejprve nutno natrénovat.

Trénovací data i hodnocení bude probíhat v simulátoru CoppeliaSim a za robota poslouží robotická ruka o sedmi kloubech.

Dílčími cíly této práce jsou:

1. Vytvořit dataset motorických akcí sloužící pro jejich naučení. Tento dataset má obsahovat různé akce a měnit parametry tělesa.
2. Připravit různé varianty reprezentace dat vstupujících do generativního modelu. Tyto vstupy se mají lišit množstvím dat a použitím příznaků.
3. Vybrat (a přizpůsobit si) vhodný generativní model a natrénovat jej na připravených datasetech.
4. Vyhodnotit a porovnat různě natrénované modely z hlediska úspěšnosti manipulace.

Kapitola 2
Přehled související literatury

V této části je nastíněn přehled literatury zabývající se přípravou datasetů, reprezentací akcí a generativními modely.

2.1 Datasety

V současnosti existuje několik datasetů obsahujících data z reálného světa, ve kterých robot manipuluje s objekty. Například Kuan-Ting-Yu a spol. [13] vytvořili dataset s posouváním objektů obsahující přes milion akcí s jedenácti různými tělesy. Dalším příkladem je dataset s nehomogenními tělesy, kde bylo použito 250 různých těles a pro každé z nich bylo zaznamenáno 250 posunutí. Tyto datasety mohou posloužit např. pro evaluaci současných algoritmů.

Pokud chceme natrénovaný model použít pro ovládání fyzického robota, je nutné sestavit si buď vlastní trénovací dataset, nebo použít simulátor a potřebná data vygenerovat. Datasetem zaměřeným na zvedání předmětů se zabývá tento článek Zhang a spol. [14]. Tyto akce již byly prováděny v simulátoru.

Náš dataset obsahuje tři druhy akcí. Pro každý druh tělesa pak máme stejný počet příkladů od každé z akcí. Další příklady je možné vygenerovat s pomocí programu, jež je součástí této práce.

2.2 Reprezentace akcí

Pousouvání se často reprezentuje pomocí vizuálního záznamu. Zhuo Xu a spol. zachycovali posouvání předmětů pomocí obrazu i přesto, že pokusy probíhaly v simulátoru [12]. Dalšími příklady mohou být projekty již citované výše [13][1].

Videozáznam je však obtížně zpracovatelný, protože obsahuje velké množství dat. Klasifikace předmětů z obrazu je navíc citlivá na světelné podmínky. Proto se řada výzkumů [2] zabývá přepisem trajektorie těles z videozáznamů do vektorů.

Pro zachycení pozice člověka se lze použít model OpenPose, který představili Zhe Cao a spol. [3]. Jedná se o reprezentaci pomocí pozic kloubů.

Přestože naším objektem zájmu není člověk ale robot, tímto přístupem jsme se inspirovali.

V této práci jsou pohyby reprezentovány v podobě úhlů kloubů robota, pozice tělesa a dalších informací 4.1.2. Výhodou této reprezentace je, že ji lze převést zpátky na pohyb.

2.3 Generativní modely

Pro generovaní akcí se používají různé metody. Jedním z příkladů je GAN, generativní adversariální síť. Základem tohoto modelu jsou dvě neuronové sítě, jedna generující a druhá hodnotící. Trénovány jsou obě současně, přičemž generativní síť se snaží vygenerovat data co nejpodobnější trénovacím datům. Cílem hodnotící je rozlišit trénovací data od vygenerovaných. Tento model byl také použit pro generování pohybů [8].

Dalším generativním modelem je ERD, tedy enkodér-rekurentní dekodér. Tento model se používá pro predikci budoucích pohybů na základě současného stavu [4].

Variační autoenkodéry (VAE) byly použity například v tomto článku [7]. Jsou tvořeny dvěma neuronovými sítěmi, pro které lze mimo jiné použít architekturu Transformer, což je architektura specializující se na sekvenční data [9].

VAE jsme si vybrali právě kvůli možnosti zpracovávat sekvenční data pomocí Transformátorů. Další výhodou VAE je, že je možné podmiňovat generovaná data např. typem akce.

Kapitola 3

Metody

V této kapitole bude představen teoretický základ práce. Nejdříve se zaměříme na prostředky k vytvoření prvního datasetu akcí. Následovat bude popis použitého modelu neuronové sítě. Nakonec budou představeny metody, jak data pro model připravovat a jak hodnotit výsledky.

3.1 Příprava datasetu

Pro naučení modelu jsou potřeba vstupní data. Tato data budou vytvářena tak, že se budou příklady jednotlivých akcí provádět v simulátoru. Do výsledného datasetu se dostanou pouze reprezentace úspěšných pokusů.

3.1.1 Definice problému

Je potřeba vytvořit dataset motorických akcí *(posuň, strč a zvedni)* s různými parametry (velikost, tvar, hmotnost). Dataset bude vytvořen v simulovaném prostředí CoppeliaSim. Dále je potřeba průběhy pokusů vhodně reprezentovat a tuto reprezentaci uložit.

3.1.2 Experimentální prostředí

V této části bude popsán simulátor Coppeliasim, knihovna umožňující práci s tímto simulátorem prostřednictvím programovacího jazyka Python a nakonec robotická ruka, která bude vykonávat akce popisované v této práci.

CoppeliaSim

Simulátor CoppeliaSim, dříve známý pod jménem V-REP, je jedním z nejrozšířenějších robotických simulátorů vůbec. Mezi jeho přednosti patří přenosnost, škálovatelnost a široká paleta využití [10]. Používá se jak pro studijní a výzkumné účely, tak také v průmyslu. Uplatní se i ve firmách jako je NASA, Honda nebo Bosh.

Pyrep

Knihovna Pyrep [5] slouží jako prostředník mezi programátorem a simulátorem CoppeliaSim. Pyrep je napsán v jazyce Python a je určen pouze pro operační systém Linux.

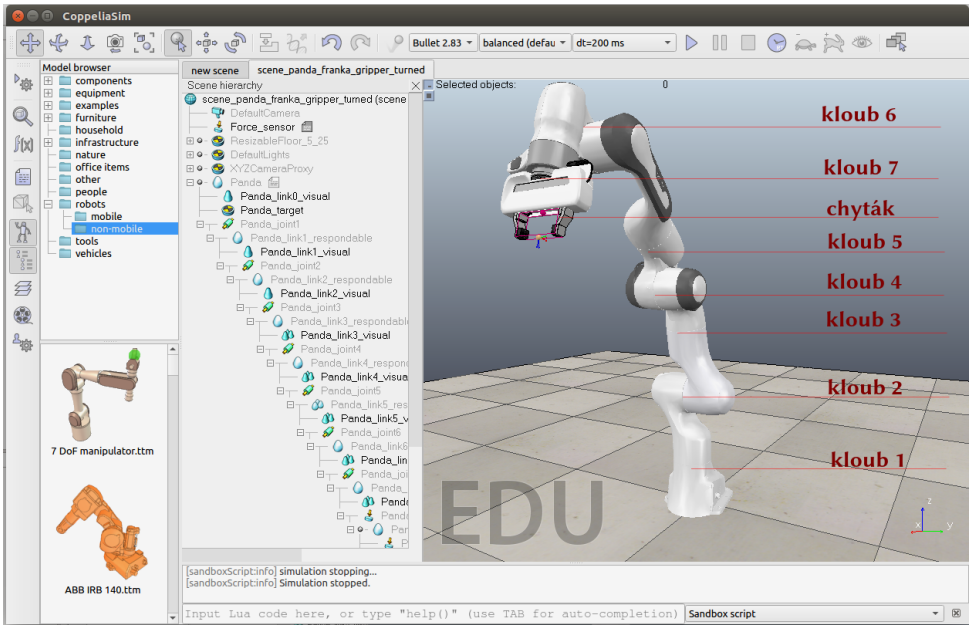
Skrze tuto knihovnu lze simulátor spustit, provést jednotlivé kroky simulace a nakonec opět zavřít. Dále zprostředkovává funkce pro vytváření a ovládání objektů. Objekty mohou být například jednoduchá tělesa, klouby nebo sensory.

V neposlední řadě Pyrep nabízí podporu pro celé roboty. V současnosti se jedná o deset různých robotických rukou a tři mobilní roboty.

Robotická ruka

Robotická ruka použitá v této práci také pochází z knihovny Pyrep. Zde se nachází pod názvem Franka Emika Panda. Jedná se o sedmikloubé rameno ukončené chytákem.

Zespoda je ruka připevněná k podložce, takže její manipulační prostor je vymezen délkou a ohebností jejích částí. V nulové výšce tak dosáhne 2,45 až 0,35 jednotek daleko od své základny.



Obrázek 3.1: Simulátor CoppeliaSim s robotickou rukou.

3.2 Generování akcí

Generování akcí bude probíhat pomocí upraveného variačního autoenkodéru. K tomuto účelu bude použita implementace ACTOR. Tato implementace již obsahuje úpravu umožňující učení se různých akcí na jednom modelu.

Dále budou představeny možnosti úpravy datasetu, které by mohly napomoci k lepším výsledkům.

3.2.1 Definice problému

Na základě příkladů akcí se chceme naučit generovat podobné příklady. Těchto akcí je několik typů a mají různé parametry.

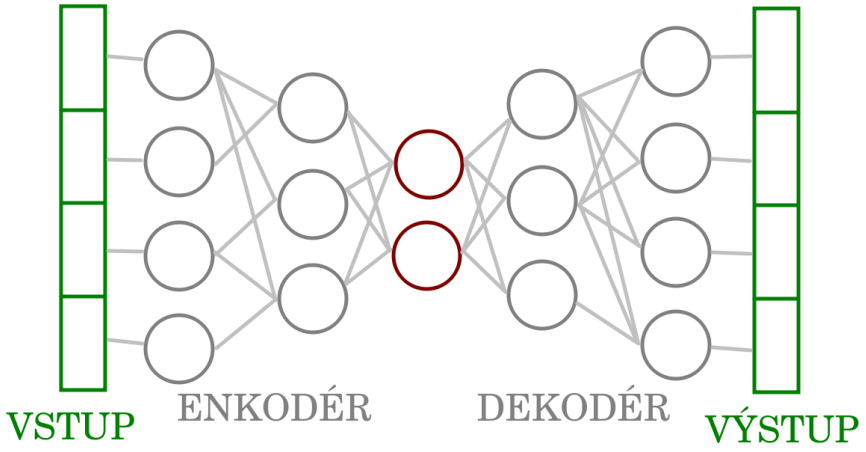
3.2.2 Variační autoenkodéry

Variační autoenkodéry jsou jedním z druhů umělých neuronových sítí určených pro generování dat. Jejich smyslem je naučit se na omezeném množství příkladů generovat další příklady, které budou těm původním podobné.

Princip práce variačních autoenkodérů vychází z obyčejných autoenkodérů. Také zde jsou proto nejprve vysvětleny autoenkodéry.

Autoenkodér

Autoenkodéry jsou určeny ke komprimaci a následné dekomprimaci dat. To se využívá například pro odstranění šumu v obrázcích. Základními částmi autoenkodéru jsou enkodér a dekodér. Schéma autoenkodéru je na obrázku 3.2.



Obrázek 3.2: Schéma autoenkodéru.

Enkodér je tvořen umělou neuronovou sítí, jejímž vstupem jsou vstupní data a výstupem je vektor dané dimenze. Prostor těchto vektorů se nazývá latentní prostor. Druhou neuronovou sítí je dekodér, který z vektoru z ekodéru data rekonstruuje.

Zúžení mezi enkodérem a dekodérem se realizuje jako tzv. bottleneck layer, tedy vrstva o nejnižším počtu neuronů.

Autoenkodér se učí po epochách. Jedná se o tzv. učení bez učitele, tedy bez nutnosti přiřazovat ke vstupu cílovou proměnnou (např. třídu). Učení

řídí ztrátová funkce, která vyhodnocuje rozdíl mezi vstupem a výstupem autoenkodéru. Na konci každé epochy se pak na základě ztrátové funkce přenastaví váhy v obou neuronových sítích.

Dekodér autoenkodéru se po natrénování chová téměř jako generátor. Bere latentní vektor na vstupu a rekonstruuje z něj vzorek původních dat (tedy například jeden pokus nebo obrázek).

Nevýhodou autoenkodéru je, že neumí zobecňovat a vytvářet nová, dosud neviděná data. Pokud by se na vstup dekodéru autoenkodéru dostal vektor, který by nepocházel z enkodéru, dekodér by z něj pravděpodobně vytvořil pouze šum. A právě zde přichází vylepšení v podobě variačních autoenkodérů.

Variační autoenkodér

Variační autoenkodér klade požadavky na uspořádání latentního prostoru. To jest, aby dva body, které jsou v latentním prostoru blízko sebe, daly podobný výstup a aby všechny vygenerované výstupy byly smysluplné. Toho se docílí tak, že se vektory v latentním prostoru převedou na distribuci s normálním rozdělením. Distribuce umožňuje pokrýt určitou část latentního prostoru. Vzorky vygenerované z tohoto prostoru by měly být na výstupu mírnou obměnou příkladu prezentovaného na vstupu.

Výstupem enkodéru variačních autoenkodérů už není vektor, ale distribuce. Tato distribuce je dána střední hodnotou a směrodatnou odchylkou. Střední hodnota i směrodatná odchylka jsou parametry, které se model učí, je tedy potřeba, aby se tomu uzpůsobila ztrátová funkce. Ta se nyní skládá nejen z porovnání vstupu a výstupu, ale obsahuje také Kullback-Leiblerovu divergenci. Ta udává rozdíl mezi parametry distribuce poskytnuté enkodérem a normálním rozdělením (kde střední hodnota je 0 a směrodatná odchylka 1). Minimalizace této divergence zajišťuje rovnoměrnější rozložení distribucí pro jednotlivé trénovací příklady napříč latentním prostorem. Je tedy podmínkou pro generování smysluplných náhodných vzorků z latentního prostoru.

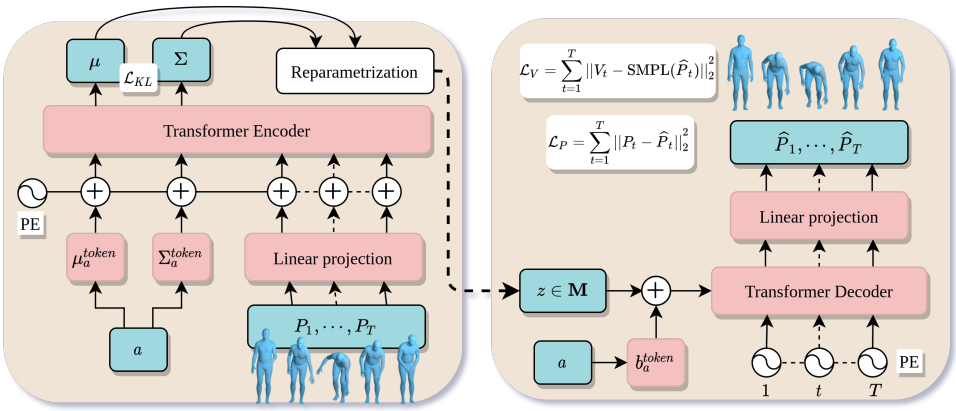
Celkově pak ztrátová funkce [6] vypadá následovně 3.1.

$$L(x, \hat{x}) + \sum_{j} KL(q_j(z|x)||p(z)) \tag{3.1}$$

kde L je původní ztrátová funkce, tedy porovnání vstupu x a výstupu $\hat{x}$. Druhá část značí Kullback-Leiblerovu funkci aplikovanou na každou z dimenzí latentního prostoru. Tato ztrátová funkce se též označuje jako ELBO (anglicky Evidence lower bound) a vyjadřuje spodní hranici maximální věrohodnosti pozorovaných dat.

ACTOR

Model ACTOR [9] je konkrétní implementací variačního autoenkodéru použitou v této práci. Původně byl vytvořen za účelem generování několika různých druhů pohybů zachycených v podobě pozic kloubů člověka. Jeho schéma je na Obr. 3.3.



Obrázek 3.3: Schéma modelu ACTOR.

Data z příkladů pohybů (P_1, ..., P_T) jsou zarovnána do tenzorů a vstupují do enkodéru. Na rozdíl od klasického VAE vstupují do enkokdéru také popisy distribucí akcí viz [9].

Při generování nových příkladů (pravá část schématu) se pak k vektoru z z latentního prostoru přičítá ještě vektor přiřazený danému druhu akce. Tímto způsobem dojde ke kontrolovanému rozmístění druhů akcí v latentním prostoru.

Použitou architekturou pro enkodér a dekodér jsou Transformery [11]. Tedy neuronové sítě s pozornostním modulem, které se specializují na zpracování sekvenčních dat.

3.2.3 Příznaky pro učení modelu

Hrubá, tedy nijak nezpracovaná, data nemusí být pro další použití ideálně čitelná. Některé důležité hodnoty totiž nemusí být vidět, byť je možné je dopočítat. Příkladem je například změna vzdálenosti mezi robotickou rukou a tělesem. Předpočítání těchto údajů, tzv. příznaků, může usnadnit učení nejen člověku, ale i stroji.

Na druhou stranu, s množstvím příznaků roste i celkový objem dat. Příliš mnoho údajů ke zpracování dělá učení náročnějším a působí tedy kontraproduktivně. Proto může pomoci vyřazení těch původních údajů, které lze z příznaků dopočítat.

Příznaky použité v této práci jsou popsány níže v Sekci 4.1.3.

3.3 Evaluace

Generovaná data je možné hodnotit pozorováním nebo měřením. Pozorování je náročnější na lidský čas, ale výhodou je komplexnější vyhodnocení. Pozorováním totiž lze podchytit i špatně měřitelné nebo nepředpokládané jevy. Zároveň je dobrým východiskem pro další vylepšení.

Výhodou měření je, že je lze automatizovat, a tedy lidský čas ušetřit. To umožňuje zhodnotit větší vzorky dat.

3.3.1 Kritéria kvality generovaných akcí

Akce popisované v této práci se zaměřují na manipulaci s tělesy. Úspěch pokusu je proto dán pohybem tělesa. V práci jsme používali následující kritéria:

- Při akci *posuň* musí ruka odtlačit těleso tak, aby se krátce po zastavení ruky také zastavilo.
- Kritériem akce *strč* je rozdíl polohy tělesa v okamžiku, kdy se naposledy dotýkalo ruky, a konečné polohy tělesa.
- Akce *zvedni* je ohodnocena podle výsledné vzdálenosti tělesa od podložky. Konkrétní hodnoty pro daná kritéria jsou popsána v sekci 4.3.2.

  Dalším, mírnějším kritériem je charakter pohybu ruky. Předmětem hodnocení by pak byla její rychlost a pozice vůči tělesu. Tento způsob hodnocení je použit pro pokusy vygenerované modelem.

Kapitola 4

Implementace

4.1 Příprava datasetu

V této části je popsána výroba příkladů akcí. Příklady akcí jsou prováděny v simulátoru, validovány a ukládány. Nakonec jsou datasety připravovány na trénování.

4.1.1 Tvorba virtuálního prostředí

Scéna

Defaultní scéna po spuštění simulátoru CoppeliaSim již obsahuje rovnou podložku, kamery a reflektory. Tato scéna byla přímo v simulátoru upravena. Do středu souřadnic (a zároveň i podložky) byla umístěna robotická ruka Franka Emika Panda. Kamera byla nastavena tak, aby snímala celou ruku i její manipulační prostor. Scénu je možné vidět na Obr. 3.1.

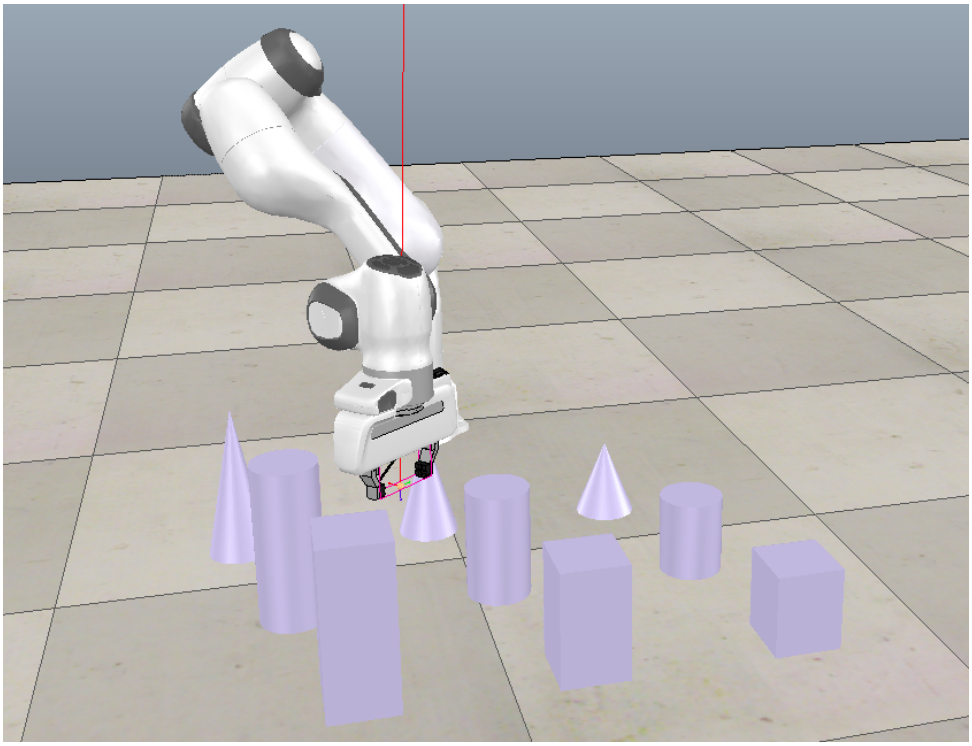
Tělesa

Na začátku každého pokusu je do scény přidáno nové těleso. Je tomu tak kvůli tomu, že jednou umístěnému tělesu nelze měnit parametry jako je hmotnost nebo tvar. Po ukončení pokusu je těleso ze scény odstraněno.

V této práci byla použita tělesa o třech různých výškách a dvou hmotnostech. Z tvarů byl použit kvádr, válec a kužel. Tato tělesa můžete vidět na obr. 4.1.

Do scény je při akci *posuň* ještě volitelně přidávána vizualizace cílového bodu. Tato vizualizace je ztvárněna tělesem nulové výšky.

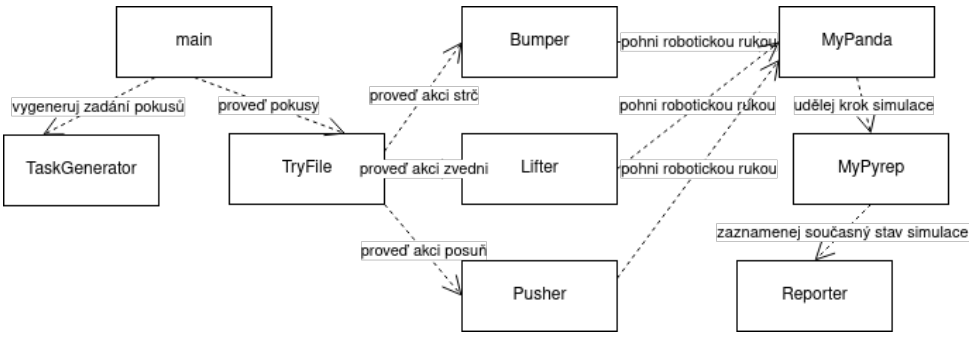
Druhá vizualizace je použita pro názornější rekonstrukci pokusů vygenerovaných naučeným modelem. Reprezentuje polohu tělesa předpočítanou modelem. Je ztvárněna tělesem o tvaru a velikosti, pro jaké byl model generován. Narozdíl od "reálného" tělesa však neinteraguje s prostředím.



Obrázek 4.1: Použitá tělesa v porovnání s robotickou rukou.

4.1.2 Generování a validace dat

V této části práce se budeme zabývat prováděním, validováním a ukládáním příkladů akcí. Na Obr. 4.2 můžeme vidět přehled tříd, které byly k tomuto účelu vytvořeny.



Obrázek 4.2: Zjednodušený diagram tříd určených pro vytváření datasetů.

Popis zadání pro robota

Posunout těleso znamená působit na ně silou tak, aby se pohybovalo smykem po podložce. Těleso by se mělo zastavit krátce po té, co se ho

robotická ruka přestane dotýkat. Má tedy najet před těleso a tlačit ho před sebou. Ke konci akce by měla zpomalit.

Strčit do tělesa znamená předat mu dostatek energie na to, aby setrvalo v pohybu i poté, co na ně přestaneme působit silou. Před kontaktem s tělesem ruka zvýší svoji rychlost. Následně jej před sebou tlačí, a to po kratší dobu než při posunutí. Nakonec ještě krátce zrychlí a náhle zastaví.

Při zvedání by ruka měla uchopit těleso zeshora a změnit jeho polohu tak, aby se dostalo nad podložku (a nedotýkalo se jí).

Průběh vykonávání pokusů

Před samotným vykonáváním pokusů jsou do JSON dokumentu vygenerovány jejich parametry. Těmito parametry jsou převážně vlastnosti tělesa, konkrétně typ objektu, rozměry, hmotnost a součinitel tření. Dále jsou zde uloženy informace týkající se konkrétního typu akce.

Ukládané vlastnosti tělesa mohou být buď náhodné z předvybraného intervalu, nebo mohou být nastaveny na konkrétní hodnoty. Intervaly i vybrané konkrétní hodnoty použité v této práci byly vybrány na základě pohybových možností robota a pozorování.

Na začátku každého pokusu se nejprve vytvoří těleso o daných vlastnostech. Následně robotická ruka provede sadu pohybů určených požadovaným druhem pohybu. Poté se případně uloží reprezentace právě ukončeného pohybu.

Po vykonání pokusu se těleso odstraní, aby nepřekáželo. Robotická ruka je však defaultně ponechána v pozici, ve které skončila. Tím se zvýší různorodost pokusů. Stává se však, že se robotická ruka dostane do pozice, ze které už není schopna provádět další pokusy. Proto je v případě nezdařilého pokusu preventivně vrácena zpět do výchozí pozice.

S větším množstvím provedených pokusů přesnost simulátoru klesá. V pravidelných intervalech je proto celý simulátor restartován.

Validace vygenerovaných dat

Ne všechny pokusy lze označit za zdařilé. Proto je každý pokus uložen pouze pokud splňuje požadavky akce, ke které má náležet.

Akce *posuň* je uložena tehdy, když se těleso od okamžiku zastavení ruky nepohne o více než 0,01 jednotky. Zároveň se těleso nesmí převrátit.

Akce *strč* je naopak úspěšná tehdy, když je táž vzdálenost, tedy od pozice tělesa těsně poté, co se ruka zastavila, k pozici, do které se těleso dostalo, větší než 0,03 jednotky.

Akce *zvedni* je úspěšná tehdy, když je na konci akce těleso alespoň 0,01 jednotek nad podložkou.

Tyto požadavky vycházejí z faktu, že je předem známo, ke kterému druhu akce má pokus náležet. (Nehodnotí se posouvání, pokud má být výsledkem zvednutí.) Také je zde využita možnost rozdělit si průběh pokusu na předem známé části. (Je jisté, že v určitém okamžiku robotická ruka zastaví, takže si můžeme pokus rozdělit na části před zastavením a po něm.)

Ukládání vygenerovaných dat

Průběhy akcí se ukládají ve formě JSON dokumentu. Každý zápis akce má hlavičku s hodnotami, které se v průběhu pokusu nemění. Následuje seznam hodnot pořizovaných po krocích simulace. Z důvodu výsledné velikosti dat se měření provádí po třech krocích simulace.

Do hlavičky zápisu se ukládají především vlastnosti tělesa. Konkrétně jde o typ tělesa (tj. kvádr, kužel, válec) hmotnost, velikost a součinitel tření. Dále jsou zde pro kontrolu pořadové číslo akce a název vykonávané akce.

Průběh akce je zaznamenáván po každých třech krocích simulace. Měří se čas, úhly všech kloubů robotické ruky, jejich rychlost, míra otevření chytáků, poloha konce ruky a poloha tělesa. Začátek dokumentu vypadá například takto 4.3.

{"samples": [{"details": { "number": 3, "action": "push", "shape": { "type": "CUBOID", "mass": 10.0, "friction": 0.05000000074505806, "size": [0.07999999821186066, 0.07999999821186066, 0.14000000059604645]}}, "steps": [{"tip position": [0.4020051956176758, -0.285106360912323, 0.22560784220695496], ...

Obrázek 4.3: Příklad uložení dat ve formě JSON.

4.1.3 Výpočet a výběr příznaků

Použili jsme dva druhy příznaků:

1. Prvními použitými příznaky byly rychlost tělesa v_t a rychlost chytáku ruky v_g. Rychlosti byly počítány jako změna polohy za poslední tři kroky simulace. Cílem bylo zvýraznit pohyb tělesa a jeho souvislost s pohybem chytáku ruky.
2. Dalším použitým příznakem byla vzdálenost horního dna (resp. špičky kužele) od podložky. Poloha tělesa je v simulátoru udávána podle středu tělesa. Robot se však k tělesu přibližuje ze shora, takže by mu mohla pomoci informace, jak nízko se musí pohnout, aby se tělesa dotknul.

   Tato hodnota byla vypočítána jako z-ová souřadnice tělesa zvětšená o polovinu výšky tělesa. Tento výpočet je založen na tom, že se nepředpokládá výraznější naklopení nebo přetočení tělesa.

Konkrétní příznaky použité v jednotlivých experimentech jsou uvedeny v sekci 4.1.3

4.1.4 Popis použitých datasetů

Aby bylo možné s daty z pokusů snadno pracovat, byla ukládána do samostatných souborů. Každý z takto vzniklých datasetů obsahuje data z jednoho druhu akce a danou kombinací vlastností tělesa. Další datasety pak vznikaly přidáváním příznaků, dělením a skládáním originálních datasetů.

Vygenerované datasety

Každý z vygenerovaných datasetů existuje ve verzi *posuň*, *strč* a *zvedni*. Při vytváření datasetů pro trénování tak lze snadno získat přesný počet pokusů od každého druhu.

Datasety s různými vlastnostmi těles jsou vytvářeny tak, aby obsahovaly stejný počet příkladů od každé z kombinací vlastností těles.

Byly vytvořeny datasety popsané v tabulce 4.1 (vše je v jednotkách simulátoru).

Název	# příkladů	Výšky	Tvary	Hmotnost
One property	3000	0,14	kvádr	10
Three size object	2250	0,08; 0,14; 0,2	kvádr	10
Size type	2250	0,08; 0,14; 0,2	kvádr, válec, kužel	10
Size type weight	2250	0,08; 0,14; 0,2	kvádr, válec, kužel	10; 15

Tabulka 4.1: Přehled originálních datasetů.

Použité podoby datasetů

Použité verze datasetů již obsahují všechny druhy akcí v jednom souboru. Obsahují požadovaný počet příkladů a případně také předpočítané příznaky.

Tyto datasety jsou již ve formě uzpůsobené pro trénování, tedy nikoliv ve formátu JSON ale pkl. Informace o vlastnostech tělesa jsou zakódovány ve zvláštním souboru, stejně jako druhy akcí.

V experimentech jsme pracovali s následujícími třemi typy dat:

1. Hrubá data: V tomto případě vypadal vektor trénovacích dat následovně:

$$D = [o_1, o_2, ..., O_7, \mathbf{t_p}, \mathbf{t_o}],$$

kde o_i jsou úhly jednotlivých kloubů, $\mathbf{t_p}$ je pozice tělesa a $\mathbf{t_o}$ je orientace tělesa.

2. Hrubá data s pozicí chytáku: V tomto případě je vektor trénovacích dat rozšířen o pozici chytáku **g**:

$$D = [o_1, o_2, ..., O_7, \mathbf{t_p}, \mathbf{t_o}, \mathbf{g}].$$

3. Hrubá data s příznaky rychlosti: Hrubá data jsou doplněna o rychlost pohybu chytáku manipulátoru v_g a pohybu tělesa v_t:

$$D = [o_1, o_2, ..., O_7, \mathbf{t_p}, \mathbf{t_o}, \mathbf{g}, v_t, v_g].$$

4.2 Trénování a úprava modelu ACTOR

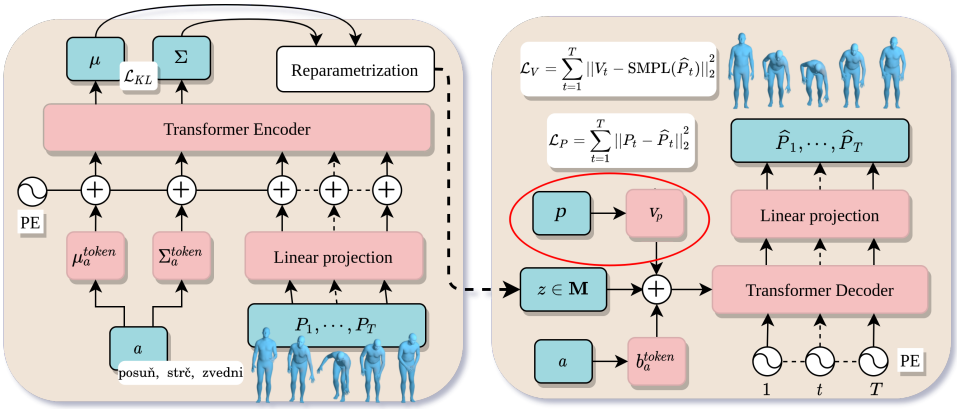
4.2.1 Úpravy v modelu ACTOR

Model ACTOR je určen pro generování lidských pohybů. Tyto pohyby reprezentuje jako úhly vybraných kloubů v jednotlivých časových úsecích. ACTOR má pevně nastavený počet kloubů. Nejprve bylo potřeba zajistit, aby se program přizpůsoboval aktuálnímu rozměru trénovaných dat.

V původním projektu se ACTOR učil generovat několik různých druhů akcí. Na žádný z těchto druhů se však nevztahovaly další parametry. Cílem této práce je generovat akce podle vlastností těles. Dalším krokem proto bylo zohlednění vlastností tělesa při trénování.

Přidání parametrů tělesa

Ve druhé části schématu modelu ACTOR se k vektoru z z latentního prostoru přičítá vektor odpovídající akce. V rámci této práce se zde přičítá ještě vektor v odpovídající kombinaci vlastností tělesa. Upravené schéma modelu ACTOR je na Obr. 4.4.



Obrázek 4.4: Upravené schéma modelu ACTOR.

Norma vektoru v nesmí být příliš velká, aby se od sebe příliš nevzdálily výsledné vektory od téhož druhu akce. Když bude norma vektoru v naopak příliš malé číslo, nenastane ve vygenerovaných příkladech dostatečná změna.

Model ACTOR používá pro posuny latentního vektoru k rozlišení akcí vektory o normě v rozmezí 7,4 až 8,3. Pro účely parametrizace akcí jsme použili vektory s normami 0,229 až 0,917.

Použité hyperparametry trénování

Pro všechna trénování v této práci byl latentní prostor omezen na 64 dimenzí. Každé trénování trvalo 5000 epoch.

4.3 Rekonstrukce akcí a vyhodnocení kvality generovaných akcí

Rekonstrukce vygenerovaných akcí se od tvorby originálních dat liší tím, že je rozdělena na velmi krátké časové úseky. Nelze tedy spoléhat na delší lineární části. Hodnocení rekonstruovaných pokusů je založeno na rozpoznání druhu akce.

4.3.1 Popis rekonstrukce akcí

Pro každou kombinaci vlastností tělesa je rekonstruován daný počet příkladů od každého z druhů akcí. Nejprve se příklady převedou z formátu pkl do slovníku. Následně je připravena scéna podobně jako při vytváření nových příkladů. Volitelně je navíc přidáno těleso, které se v průběhu pokusu bude chovat podle odhadu modelu.[1]

Dále je pohyb rekonstruován po jednotlivých krocích simulace. Nakonec je vyhodnoceno, kterému z druhů akcí se pokus nejvíce podobal. Výsledek je zaznamenán do matice záměn pro danou kombinaci vlastností. Vedle matice záměn se ještě počítají pokusy, při kterých se těleso nepohnulo.

4.3.2 Kritéria akcí

Každý pokus je ohodnocen třemi čísly, která určují podobnost ke každému z druhů akcí. Jako výsledný druh akce se pak bere ten s nejvyšším číselným ohodnocením.

Součástí hodnocení je nejen výsledný pohyb tělesa, ale také charakter pohybu ruky. Lze tak zohlednit i pohyby, které byly vykonány až na určité

[1] Zde je dobré si uvědomit, že model dostal při učení matici hodnot, kde nebylo rozlišeno, jedná-li se o souřadnici tělesa nebo úhel některého kloubu. Odhad polohy tělesa se proto může zdát zvláštní.

nepřesnosti správně. Dalším důvodem je pak omezení vlivu nepřesností simulátoru a reprodukce pohybu. Hodnocení se ověřovalo na původních datech.

Kritérii jsou:

- zvednutí tělesa - Toto kritérium je důležité hlavně pro akci *zvedni*.
- vzdálenost tělesa od ruky v době, kdy se těleso pohybuje - Rozlišuje mezi akcemi *posuň* a *strč*. U akce *posuň* by se vzdálenost mezi chytákem a tělesem neměla příliš měnit. Akce *strč* naopak požaduje, aby se po nějaké době těleso od chytáku vzdálilo.
- poloha chytáku ruky oproti tělesu - Při akci *posuň* by se ruka měla dostat těsně za těleso, u akce *strč* by navíc měla začít dál od tělesa. U akce *zvedni* se ruka blíží k tělesu ze shora.
- maximální rychlost ruky - Při akci *posuň* by ruka neměla být tak rychlá jako při akci *strč*.

Kapitola 5

Experimentální výsledky

První trénování a hodnocení byla prováděna na příkladech s tělesem o stále stejných vlastnostech. Teprve potom se trénovaly a hodnotily modely naučené na různé velikosti tělesa, následně se měnil také tvar a nakonec hmotnost. Obdobně probíhala práce s příznaky a různými velikostmi datasetů. Matice záměn jsou počítány pro 100 vygenerovaných příkladů od každé akce.

Špatně vyhodnocené akce byly často označovány jako akce *posuň*. Je to dáno tím, že jak špatně provedená akce *zvedni*, tak špatně provedená akce *strč* se akci *posuň* podobají. Nepovedlo-li se robotické ruce těleso zvednout, tak obvykle proto, protože o něj zavadila. Tím těleso posunula a zvedla se sama. Nepovedené strčení typicky selhalo tím, že bylo příliš slabé. A slabé strčení má k posunutí velmi blízko.

Při vyhodnocování akcí vygenerovaných modelem se stávalo, že se ruce vůbec nepodařilo tělesa dotknout. V tom případě byl pohyb vyhodnocen jako některá z akcí na základě svého charakteru. Vzhledem k tomu, že se jedná o poměrně velký nedostatek, uvádím počet těchto pokusů jako čtvrtý sloupec matice záměn.

5.1 Pro jeden druh tělesa

Nejprve byly hodnoceny pokusy, které byly generovány pro jeden typ tělesa. Tímto tělesem byl kvádr o výšce 1,4 jednotek a hmotností 10 jednotek. K výsledkům z těchto hodnocení pak bylo přihlíženo při trénování a hodnocení dalších datasetů.

Posun latentního vektoru pro jednotlivé akce činil 7,4 až 8,3. Tyto hodnoty byly ponechány i pro ostatní modely.

5.1.1 Velikost datasetu

Prvním objektem zkoumání byl vliv velikosti datasetu na generovaná data. Celkem byly natrénovány čtyři modely na datasetech obsahujících 100, 250, 500 a 750 příkladů od každé akce.

Na datasetu o 100 příkladech od každé akce už se model vcelku dobře naučil s rukou hýbat, ale na dobré napodobení akcí to nestačilo. Matice záměn je ukázána v Tab. 5.1.

generováno\vyhodnoceno	*posuň*	*strč*	*zvedni*	těleso se nepohnulo
posuň	59	41	0	0
strč	33	67	0	11
zvedni	33	0	67	2

Tabulka 5.1: Matice záměn pro dataset s 300 příklady a jedním typem tělesa.

Model naučený na 250-ti příkladech od každé akce dopadl výrazně lépe (viz. Tab 5.2). Předpokládaná poloha tělesa příliš neodpovídala realitě. Podle modelu se těleso mělo často samo pohybovat a vznášet se nad podložkou. I přesto byly reprodukované akce relativně úspěšné.

generováno\vyhodnoceno	*posuň*	*strč*	*zvedni*	těleso se nepohnulo
posuň	99	1	0	14
strč	6	92	2	3
zvedni	18	0	82	0

Tabulka 5.2: Matice záměn pro dataset s 750 příklady a jedním typem tělesa.

Když jsme model natrénovali na 500 příkladech od každé akce, odhad polohy tělesa zlepšil. Přestože úspěšnost generovaných akcí je podobná jako pro 200 příkladů na akci (viz. Tab. 5.3), pohyby robota byly plynulejší.

generováno\vyhodnoceno	*posuň*	*strč*	*zvedni*	těleso se nepohnulo
posuň	97	3	0	14
strč	27	67	6	0
zvedni	14	0	86	0

Tabulka 5.3: Matice záměn pro dataset s 1500 příklady a jedním typem tělesa.

Konečně po 750-ti příkladech od každé akce byly pohyby takřka stejně dobré jako ty originální (viz. Tab. 5.4). Pouze výjimečně jsme dostali sérii několika generovaných pohybů, které se od originálních výrazně lišily. Stále občas došlo k neúspěšnému pokusu o zvednutí, kdy se nepodařilo uchopit těleso.

5.1.2 Použití příznaků

Pozice chytáku ruky stojí na pomezí příznaků a hrubých dat. Je získávána přímo ze simulace, ale je ji také možné spočítat z úhlů kloubů. Pro porovnání byl tedy natrénován jeden model bez použití souřadnic chytáku. Výsledky pro jednotlivé akce jsou vidět v Tabulce 5.5.

generováno\vyhodnoceno	*posuň*	*strč*	*zvedni*	těleso se nepohnulo
posuň	100	0	0	9
strč	7	93	0	0
zvedni	37	0	63	0

Tabulka 5.4: Matice záměn pro dataset s 2250 příklady a jedním typem tělesa.

generováno\vyhodnoceno	*posuň*	*strč*	*zvedni*	těleso se nepohnulo
posuň	100	0	0	4
strč	30	70	0	0
zvedni	47	0	53	0

Tabulka 5.5: Matice záměn na datech trénovaných pouze s klouby ruky a pozicí a orientací tělesa.

Při rekonstrukci pokusů předpokládaná pozice tělesa často kolidovala s rukou a ruka se hůře strefovala do tělesa.

Pro těsnější propojení ruky a tělesa byly přidány rychlosti konce ruky a tělesa. Tyto příznaky už však modelu nijak nepomohly (viz. Tabulka 5.6).

generováno\vyhodnoceno	*posuň*	*strč*	*zvedni*	těleso se nepohnulo
posuň	88	12	0	0
strč	6	94	0	0
zvedni	50	0	50	0

Tabulka 5.6: Matice záměn pro dataset trénovaný na 2250 příkladech s příznaky rychlosti ruky a rychlosti tělesa.

5.2 Více druhů těles

Pro trénování a generování příkladů s více druhy těles byla použita parametrizace, jak bylo popsáno v tabulce 4.2.1. První měněnou vlastností těles byla jejich výška. Podle výsledků parametrizace na těchto příkladech byl nastaven posun latentního vektoru pro další datasety. Ty pak obsahovaly tělesa o různých výškách a tvarech, poslední dataset zahrnoval i tělesa o různých hmotnostech.

5.2.1 Tělesa o různé výšce

Prvním měněným parametrem byla výška tělesa, a to z toho důvodu, že je na rekonstrukci nejlépe vidět. Byly použity tři různé výšky těles o hodnotách 0,1; 0,14 a 0,2 jednotky.

Zde byly také vyzkoušeny tři různé varianty posunů vektoru z z latentního prostoru Výsledky evaluace jsou shrnuty v Tabulce 4.2.1).

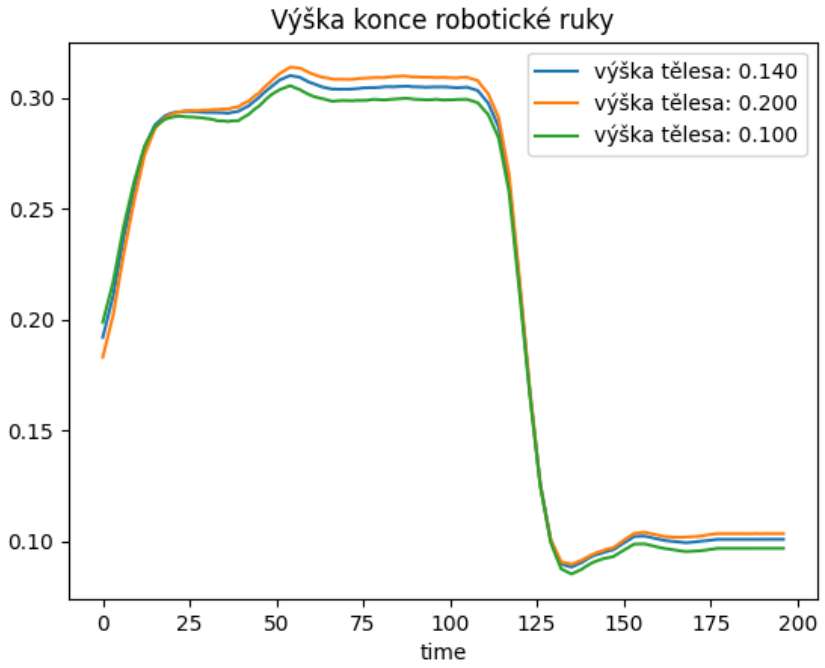
Konkrétně šlo o posuny s normami 0,229; 0,458 a 0,917. Výsledky modelu s nejmenším posunem zachycuje tabulka 5.7.

generováno\vyhodnoceno	(výška)	*posuň*	*strč*	*zvedni*	statické těleso
posuň	0,1	100	0	0	43
	0,14	100	0	0	36
	0,2	91	9	0	0
strč	0,1	33	67	0	32
	0,14	2	98	0	0
	0,2	3	97	0	0
zvedni	0,1	69	0	31	69
	0,14	40	4	56	0
	0,2	81	7	12	0

Tabulka 5.7: Matice záměn pro různě velká tělesa při posunu latentního vektoru s normou 0,229.

Výška chytáku ruky se měnila mnohem méně, než v originálních datech, jak je zachyceno na grafu 5.1.

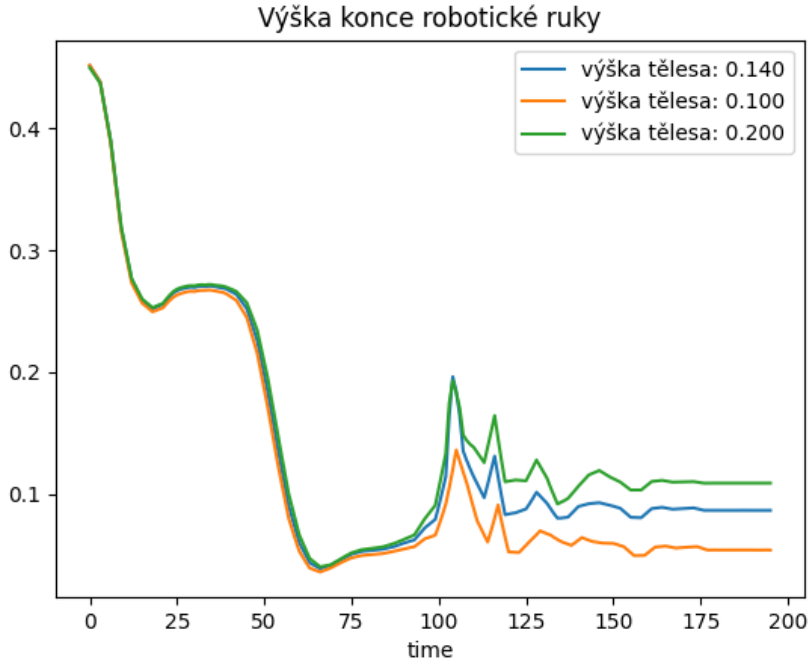
Na tomto grafu je zachycen pohyb ruky při akci *posuň*. Ruka se nejprve zvedá, aby do tělesa nenarazila příliš brzy. Následně se přesune za těleso a klesá. Právě výška, do které se v této chvíli spustí, je stěžejní, protože pokud zůstane příliš vysoko, může těleso minout.



Obrázek 5.1: Graf závislosti výšky chytáku ruky na čase pro akci *posuň* s posunem latentního vektoru o normě 0,229.

generováno\vyhodnoceno	(výška)	*posuň*	*strč*	*zvedni*	statické těleso
posuň	0,1	100	0	0	19
	0,14	92	8	0	18
	0,2	99	0	1	0
strč	0,1	23	77	0	0
	0,14	3	98	1	0
	0,2	2	96	0	0
zvedni	0,1	48	0	52	15
	0,14	64	0	36	0
	0,2	97	3	0	7

Tabulka 5.8: Matice záměn pro různě velká tělesa při posunu latentního vektoru o vektor s normou 0,458. Model byl trénován na datasetu o 2250 příkladech.

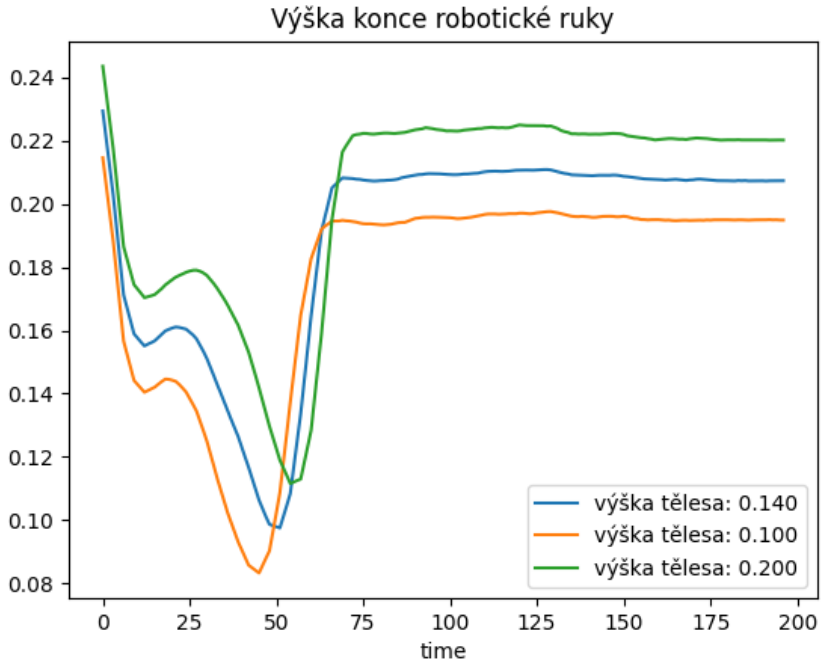


Obrázek 5.2: Graf závislosti výšky chytáku ruky na čase pro akci *strč* s posunem latentního vektoru o normě 0,458.

Dále byl natrénován model s posunem latentního vektoru o vektor s normou 0,458 (viz. Graf 5.8. Zde již byly při některých reprodukovaných pokusech patrné větší rozdíly oproti originálním datům. Při některých pokusech se ruka pohybovala níž, než by bylo třeba u nejmenšího tělesa, jindy zůstala nad tělesem. Mezi variantami téhož pokusu pro různé výšky těles už byl při některých pokusech větší rozdíl. Příkladem je Graf 5.2 pro akci *strč*.

Nejlepšího přizpůsobení bylo dosaženo s posunem latentního vektoru o vektor s normou 0,917. Při generování byl opět posunut více, než

při trénování, aby byly rozdíly výraznější. Výsledek je vidět na grafu 5.3. Tento graf popisuje výšku konce ruky při akci *zvedni*. Matice záměn pro tento model je v Tabulce 5.9.



Obrázek 5.3: Graf závislosti výšky konce robotické ruky na čase při akci *zvedni*. Byl vygenerován pro tělesa o různých výškách za posunu latentního vektoru o vektor s normou 0,917 po natrénování na 2250 příkladech.

generováno\vyhodnoceno	(výška)	*posuň*	*strč*	*zvedni*	statické těleso
posuň	0,1	100	0	0	40
	0,14	100	0	0	10
	0,2	100	0	0	0
strč	0,1	6	87	7	0
	0,14	11	89	0	0
	0,2	0	100	0	0
zvedni	0,1	36	0	63	2
	0,14	68	0	32	0
	0,2	94	2	4	0

Tabulka 5.9: Matice záměn pro různě velká tělesa při posunu latentního vektoru o vektor s normou 0,917 po natrénování na 2250 příkladech.

5.2.2 Tělesa o různé výšce a tvaru

Tento model se měl přizpůsobit nejen změnám výšky tělesa, ale také jejich tvaru. Byl použit kvádr, válec a kužel. Latentní vektor byl

posouván o vektor s normou 0,328.

Od každé z různých kombinací vlastností tělesa a druhů akcí bylo vyhodnoceno 34 pokusů. Výsledky byly sečteny a přepočítány na procenta viz Tabulka 5.10.

generováno\vyhodnoceno	*posuň*	*strč*	*zvedni*	statické těleso
posuň	99	0	1	58
strč	11	89	0	10
zvedni	54	0	46	16

Tabulka 5.10: Matice záměn pro tělesa o různé výšce a tvaru při posunu latentního vektoru o vektor s normou 0,328.

5.2.3 Tělesa o různé výšce, tvaru a hmotnosti

Tělesa v pokusech pro toto trénování měla hmotnost 10 a 15 jednotek, opět tak, aby všechny možné kombinace vlastností byly stejně četné. Celkový počet originálních příkladů byl 2250. Od každé kombinace vlastností tělesa a akcí bylo vyhodnoceno 34 příkladů, matice záměn je tedy opět v procentech.

Byly natrénovány dva modely s posunutími latentního vektoru o vektory s normami 0,4 a 0,8. Jak můžeme vidět v tabulkách 5.11 a 5.12, model s větším posunutím latentního vektoru byl úspěšnější.

generováno\vyhodnoceno	*posuň*	strč	*zvedni*	statické těleso
posuň	100	0	0	0
strč	34	64	2	24
zvedni	84	0	16	40

Tabulka 5.11: Matice záměn pro tělesa o různé výšce, tvaru a hmotnosti při posunu latentního vektoru o vektor s normou 0,4.

generováno\vyhodnoceno	*posuň*	strč	*zvedni*	statické těleso
posuň	100	0	0	1
strč	33	66	1	9
zvedni	70	0	30	29

Tabulka 5.12: Matice záměn pro tělesa o různé výšce, tvaru a hmotnosti při posunu latentního vektoru o vektor s normou 0,8.

5.3 Diskuse výsledků

Modely VAE dobře vystihly charakter pohybů, to jest jejich tvar a rychlost. To je vidět na dobrých výsledcích generovaných akcí v případě manipulace s jedním typem tělesa (viz. Tab. 5.2 a Tab. 5.4).

Naopak potíže činily situace, kdy byla dána přesná hranice. Příkladem je pohyb nad versus pod hranicí horní hrany tělesa. Pravděpodobně to vychází z faktu, že cílem VAE je všechny hodnoty reprezentovat ve své latentní vrstvě spojitě.

Parametrizace pomocí velikosti tělesa měla na jednotlivé akce rozdílné účinky. U akcí *strč* a *zvedni* vyvolala parametrizace větší změny ve výšce, ve které se pohyboval chyták, než u akce *posuň*. Akce *posuň* měnila spíše y-ovou souřadnici chytáku.

Posuny latentního vektoru menší než asi 0,3 měly na vygenerované pokusy malý vliv. Při zvětšování posunu pak postupně docházelo k prolínání akcí, kvůli čemuž se vygenerované pokusy přestávaly podobat originálním.

Možným řešením by bylo zvětšovat nejen posuny parametrů, ale také akcí. V některých případech by také mohlo pomoci zmenšení rozptylu latentních vektorů, z nichž jsou pokusy generovány. Pokusy ze začátku generování totiž bývají viditelně horší kvality.

Kapitola 6

Závěr

V simulátoru CoppeliaSim byl vytvořen generátor dat pro učení tří typů akcí prováděných pomocí robotického manipulátoru. Jedná se o akce posuň, zvedni a strč do tělesa. Generátor umožňuje měnit parametry generovaných těles (velikost, hmotnost, tvar a tření). Dále se při generování dat mění počáteční poloha tělesa a robotického manipulátoru, který pohyby provádí.

Vytvořený dataset obsahuje kloubové souřadnice manipulátoru a pozice objektů včetně zadaných parametrů. Může být dále doplněn o další příznaky, jako například rychlost pohybu manipulátoru, vzdálenost manipulátoru od tělesa apod.

Byl upraven model ACTOR, který implementoval variační autoenkodér (VAE) pro úlohu generování jednotlivých typů pohybu. Nově se tak učil nejen různé typy pohybů, ale také upravil provedení pohybů dle konkrétních vlastností manipulovaných objektů (např. podle tvaru nebo velikosti). Tento upravený model byl natrénován a hodnocen na jednotlivých datasetech, které byly vytvořeny v první části práce.

Literatura

[1] Maria Bauza, Ferran Alet, Yen-Chen Lin, Tomas Lozano-Perez, Leslie Kaelbling, Phillip Isola, and Alberto Rodriguez. Omnipush: accurate, diverse, real-world dataset of pushing dynamics with RGB-D video. 10 2019.

[2] Oliver Beyer, Sascha Griffiths, and Philipp Cimiano. Towards Action Representation within the Framework of Conceptual Spaces: Preliminary Results. 2012.

[3] Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh. OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields. *CoRR*, abs/1812.08008, 2018.

[4] Katerina Fragkiadaki, Sergey Levine, Panna Felsen, and Jitendra Malik. Recurrent Network Models for Human Dynamics, 2015.

[5] Stephen James, Marc Freese, and Andrew J. Davison. PyRep: Bringing V-REP to Deep Robot Learning. *arXiv preprint arXiv:1906.11176*, 2019.

[6] Jeremy Jordan. Variational autoencoders., Jul 2018.

[7] Hung Yu Ling, Fabio Zinno, George Cheng, and Michiel van de Panne. Character controllers using motion vaes. *ACM Trans. Graph.*, 39(4), 2020.

[8] Qianhui Men, Hubert P. H. Shum, Edmond S. L. Ho, and Howard Leung. Gan-based Reactive Motion Synthesis with Class-aware Discriminators for Human-human Interaction. *CoRR*, abs/2110.00380, 2021.

[9] Mathis Petrovich, Michael J. Black, and Gül Varol. Action-Conditioned 3D Human Motion Synthesis with Transformer VAE. In *International Conference on Computer Vision (ICCV)*, 2021.

[10] E. Rohmer, S. P. N. Singh, and M. Freese. Coppeliasim (formerly V-REP): a Versatile and Scalable Robot Simulation Framework. In *Proc. of The International Conference on Intelligent Robots and Systems (IROS)*, 2013.

[11] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[12] Zhuo Xu, Wenhao Yu, Alexander Herzog, Wenlong Lu, Chuyuan Fu, Masayoshi Tomizuka, Yunfei Bai, C. Karen Liu, and Daniel Ho. COCOI: Contact-aware Online Context Inference for Generalizable Non-planar Pushing, 2020.

[13] Kuan-Ting Yu, Maria Bauza, Nima Fazeli, and Alberto Rodriguez. More than a Million Ways to Be Pushed: A High-Fidelity Experimental Data Set of Planar Pushing. 04 2016.

[14] Hanbo Zhang, Deyu Yang, Han Wang, Binglei Zhao, Xuguang Lan, Jishiyu Ding, and Nanning Zheng. REGRAD: A Large-Scale Relational Grasp Dataset for Safe and Object-Specific Robotic Grasping in Clutter, 2021.