Univerzita Hradec Králové

Fakulta informatiky a managementu

Katedra informačních technologií

Tvorba počítačové hry a její řešení pomocí umělé inteligence

Bakalářská práce

Autor práce: Jiří Dostál

Studijní obor: Aplikovaná informatika

Vedoucí práce: Ing. Milan Kořínek

Hradec Králové 2024

Prohlášení

Prohlašuji, že jsem bakalářskou práci zpracoval samostatně, s použitím uvedené literatury. Umělá inteligence byla v práci použita pouze pro stylistické účely.

Všechny zdrojové kódy jsou k dispozici na platformě *github*, pod odkazem `https://github.com/thevidko/Autonomous-Car-UE5`.

V Hradci Králové dne 21. dubna 2024 Jiří Dostál

Poděkování

Tímto děkuji Ing. Milanovi Kořínkovi, vedoucímu bakalářské práce, za skvělé vedení a zájem, který mi věnoval při tvorbě této práce.

Anotace

Bakalářská práce se zaměřuje na vytvoření počítačové hry a následně její řešení pomocí umělé inteligence. Hlavním cílem bylo začlenit do hry strojové učení. Hra byla vytvořena pomocí Unreal Engine a soustředí se na závodní trať, kterou projíždí autonomní závodní vůz řízený umělou inteligencí.

První přístup k implementaci strojového učení byl proveden přímo v rámci Unreal Engine bez použití jakýchkoli zásuvných modulů třetích stran. U druhého přístupu byl pro trénování a implementaci neuronové sítě využit framework PyTorch. Výsledky práce demonstrují funkčnost obou přístupů. Závěrečná část shrnuje dosažené výsledky a diskutuje možnosti dalšího výzkumu v oblasti herní umělé inteligence.

Anotation

Title: Creating of a video game and its solution using artificial intelligence

The bachelor thesis focuses on the creation of a computer game and its subsequent solution using artificial intelligence. The main goal was to incorporate machine learning into the game. The game was created using the Unreal Engine and focuses on a race track that is driven by an autonomous racing car controlled by artificial intelligence.

The first approach to implementing machine learning was done directly within the Unreal Engine, without the use of any third-party plugins. For the second approach, the PyTorch framework was used to train and implement the neural network. The results of this work demonstrate the functionality of both approaches. The final section summarizes the results and discusses the possibilities for further research in the field of game AI.

Klíčová slova

počítačové hry, herní enginy, umělá inteligence, neuronové sítě, Unreal Engine, PyTorch

Keywords

computer games, game engines, artificial intelligence, neural networks, Unreal Engine, PyTorch

Obsah

Seznam obrázků

1 Úvod

V posledních několika desetiletích se počítačové hry staly fenoménem, který zaujal miliony lidí po celém světě. Herní průmysl se vyvinul v jedno z nejlukrativnějších a nejvlivnějších odvětví na světě. Není to jen forma zábavy, ale také oblast velkého technologického pokroku a tvůrčího potenciálu. [1]

Vývoj počítačových her ušel od dob Pongu a Space Invaders dlouhou cestu. S tím, jak se v průběhu let vyvíjely počítačové hardwarové a softwarové technologie, dokázali vývojáři her vytvářet propracovanější herní zážitky. Dnešní hry se vyznačují úžasnou grafikou a realistickou fyzikou.

Umělá inteligence hraje při vývoji počítačových her stále důležitější roli. Používá se nejen při vytváření herních postav a prostředí, ale také při návrhu a mechanice samotných her. Algoritmy umělé inteligence lze využít k vytváření náročnějších protivníků, generování dynamických a nepředvídatelných herních scénářů, a dokonce i k přizpůsobení hry individuální úrovni dovedností a hernímu stylu hráče. [2]

Herní průmysl navíc není jen o zábavě. Stal se také významnou hnací silou technologických inovací. Poptávka po vysoce výkonném hardwaru a softwaru vedla k vývoji nových technologií, jako je virtuální a rozšířená realita, které se nyní využívají v různých odvětvích, od zdravotnictví po vzdělávání. S pokračujícím technologickým pokrokem a rostoucí sofistikovaností umělé inteligence se bude herní průmysl v příštích letech jistě dále vyvíjet a rozšiřovat.

Bakalářská práce se zaměřuje na průnik počítačových her a umělé inteligence. Zabývá se tím, jak lze tyto dva obory propojit a jaké potenciální přínosy z takové kombinace získat. Praktická stránka práce zahrnuje implementaci technik strojového učení do nově vytvořené počítačové hry.

Začlenění strojového učení do nově vytvářené počítačové hry zahrnuje možné nahrazení stávajících systémů, které nejsou dostatečně schopné učit se z chování hráčů a přizpůsobovat se jejich akcím. Tato implementace umožňuje personalizovanější a náročnější herní zážitek, při kterém umělá inteligence dokáže přizpůsobit úroveň obtížnosti na základě dovedností hráče.

Pro vývoj počítačové hry byl zvolen volně dostupný herní engine Unreal Engine. To značně zjednodušuje část procesu vývoje hry. Navíc představuje možnosti implementace strojového učení speciálně pro tento herní engine. Na základě těchto implementací lze strojové učení dále rozšiřovat a zdokonalovat pro budoucí herní projekty nebo pro vývoj frameworků strojového učení.

2 Cíl práce

Jak už z názvu bakalářské práce vyplývá, hlavním úkolem je vytvořit počítačovou hru a následně ji vyřešit umělou inteligencí. Pro dosažení cíle je potřeba splnit následující podcíle:

- Vytvořit počítačovou hru
- Připravit počítačovou hru pro umělou inteligenci
- Navrhnout způsob implementace strojového učení
- Naučit umělou inteligenci hrát hru

3 Počítačové hry

Za první počítačovou hru se obecně považuje hra *Spacewars*! z roku 1962. Jejím prvním hráčem a tvůrcem byl Steve Russell [3]. Od té doby se počítačové hry staly běžnou součástí života a zábavy. Zažily enormní vývoj, co se týče technologií, příběhů, grafiky a dostupnosti.

3.1 Definice

Počítačovou hrou je nazývaný interaktivní program spustitelný na osobních počítačích. Počítačové hry spadají do kategorie videoher a jsou určeny pro osobní počítače. Mezi videohry patří také mobilní hry, konzolové hry a podobně. Přes to se videohry často označují jako počítačové hry. Primární funkcí videoher je zábava. Mezi další funkce může patřit rozvoj myšlení nebo dokonce učení.

Rozdělení videoher podle zařízení:

- Herní automaty
- Pro stolní počítače a laptopy
- Pro mobilní zařízení
- Pro herní konzole

Pro zjednodušení definování počítačových her je potřeba i zmínit jejich varianty podle počtu hráčů.

- Hry pro jednoho hráče - *singleplayer*
- Hry pro více hráčů - *multiplayer*

Multiplayer hry lze rozdělit na další dva druhy, a to lokální a síťové. V případě lokální hry pro více hráčů se hraje na jednom zařízení. Tato varianta byla dříve velmi populární a samotné hry podporovaly hraní skrz speciální funkce.

S rozšířením internetu se začala rozšiřovat i varianta síťového hraní. Hráči se se svým zařízením připojují na herní servery. Dnes tento druh dominuje v multiplayer hraní.

3.2 Prvky

Počítačové hry obsahují podobné, nebo i dokonce společné prvky. Jedná se o základní rysy her, které jsou důležité pro jejich definici a odlišení od jiných forem zábavy. Tyto rysy dělají hry interaktivními, zábavnými a smysluplnými.

3.2.1 Cíl

Každá hra obsahuje svůj cíl, kterého se hráč snaží dosáhnout. Bez cíle ztrácí hra smysl. Cíl nemusí být jediný. Může existovat více cílů, které mohou na sebe navazovat, nebo nemusí mít vůbec žádnou spojitost. Existují také hry, které neobsahují pevně daný cíl a které lze po

splnění hlavního cíle dále hrát. V takovém případě je na hráči, aby si vytvořil svůj vlastní cíl, kterého chce dosáhnout.

Pro lepší pochopení lze uvést dva příklady her. Prvním typem je hra orientovaná na příběhovou linku, kde je pevně daný počet misí, které na sebe navazují a splněním poslední hra skončí. Zde je cíl pevně daný a hráč se ho drží. Příkladem takové hry je *Mafia: The City of Lost Heaven*, česká hra orientovaná primárně na příběh.

Druhým příkladem je *sandbox* hra, která dává hráči k dispozici širokou škálu funkcí, které může kombinovat a využívat k tomu vlastní kreativitu. *Minecraft* je ukázkový příklad hry, ve kterém si sám uživatel určuje, čeho chce dosáhnout a stanovený cíl není tak důležitý.

3.2.2 Odměna

Se splněním cíle souvisí i odměna pro hráče, která je jeho hlavní motivací. Odměny můžou být rozděleny na dvě různé formy.

Hráč po splnění mise obdržel herní předmět, který posílí jeho postavu, a další postup bude tak pro něho jednodušší. V tomto případě se jedná o odměnu „fyzickou“, při které hráč získává různé vybavení, herní měny, body pro rozvoj jeho postavy a další jiné motivační předměty.

Druhým případem je hráč slavící vítězství nad nepřítelem v on-line hře. Psychická odměna je nedílnou součástí tohoto typu her, při kterých hráči porovnávají síly mezi sebou. Ovšem to se netýká pouze on-line her.

3.2.3 Hraná postava

Díky rozmanitosti počítačových her nelze plně definovat hranou postavu. V tomto případě je lze rozlišit na základně závislosti na herním příběhu.

Závislé na herním příběhu

Je stanovena hlavní postava, která provází celou hrou. Její identita nemusí být známá, ale může být charakterizována podle jejích vlastností. Celý příběh se odehrává právě kolem této postavy. Samozřejmě postava nemusí být jediná hratelná, například ve hře *Grand Theft Auto V* je hlavní příběh orientován kolem tří postav, které se střídají během hraní.

Nezávislé na herním příběhu

Ve hře může být hratelná postava, ale její charakter není pro příběh hry zásadní. Hra tedy může obsahovat nepodstatnou postavu za kterou hráč hraje, více postav, z nichž si vybírá, nebo vůbec žádnou postavu.

Ve hrách pro více hráčů si hráči často vybírají postavu na základě jejích jedinečných vlastností, které ovlivňují hru nebo její vizuální stránku.

Jedním z příkladů hry bez postavy jsou strategické hry, ve kterých se hra ovládá z pohledu shora (God view), například ovládání celé populace města.

3.2.4 Svět a NPC

I herní svět se velmi odvíjí od druhu hrané počítačové hry. Herní svět slouží k umístění veškerého dění hry. Zpravidla herní svět obsahuje povrch a k němu prvky spojené (stromy, vodní plochy, domy), prvky nespjaté s povrchem, pomocné prvky napomáhající vizuální stránce a nakonec nehratelné postavy. Herní světy lze rozdělit podle možnosti modifikace tohoto světa.

Prvním druhem jsou světy, které byly přesně vymodelovány buď na základě reality, předlohy nebo fantazie. Všechny ozdobné prvky jsou vkládány ručně. Interakce se světem může do jisté míry probíhat prostřednictvím prvků, které nejsou pevně svázány s půdou světa, jako jsou vozidla nebo zničitelné objekty.

V druhém případě jsou světy trvale modifikovatelné. Jejich povrch se mění v průběhu hraní a musí být uložen. Právě tyto světy jsou často náhodně generovány pomocí algoritmů. Příkladem může být již zmíněná hra *Minecraft*, ve které existuje obrovské množství variant světů, které lze během hraní stále upravovat.

NPC, zkratka pro anglické *non-playable character* nebo česky nehratelná postava, je postava obsažená ve hře, která není nijak ovládaná hráčem [4]. Primárně fungují na základě několika pravidel, ale i tak se jejich chování označuje jako umělá inteligence. Každé NPC se odvíjí od hrané hry. Někde se používají pro doplnění světa a jejich interakce je minimální. Jinde zase jejich interakce tvoří podstatu hry.

3.3 Žánry

Dřívější počítačové hry bylo mnohem jednodušší rozdělit podle žánrů, než je tomu dnes. Producenti častokrát experimentovali a tím vytvořili různé subžánry a zcela jiné žánry. Proto bude představeno několik nejoblíbenějších žánrů dnešní doby [5].

Sandbox

Žánr obsahující velké množství funkcí s otevřeným světem. Hráči dává kompletní svobodu v tvoření. Může být obsažena i dějová linka, ale hráč si tvoří svůj vlastní zážitek podle svého rozhodnutí. Tento žánr her je často doporučován pro děti, protože podporuje jejich kreativitu. [6]

Strategie v reálném čase

Úkolem hráče je v reálném čase probíhajícího světa strategicky myslet a rozhodovat se. Zpravidla se jedná o management zdrojů, plány a racionální rozhodování.

Střílečky

Tento žánr se primárně pohybuje kolem útočných střelných zbraní. Po hráči vyžaduje přesnost střelby a taktického myšlení. Žánr je rozdělován na několik subžánrů.

FPS, neboli *first person shooter*, je kamera reprezentací přímého pohledu hrané postavy. Tento subžánr vyžaduje od uživatele nízkou reakční dobu a přesné míření.

TPS, zkratka pro *third person shooter*, je subžánr, při kterém je kamera umístěna nad postavou hráče. Větší důraz je kladen na prostředí a taktiku.

MOBA

Multiplayer Online Battle Arena nemá velké zastoupení v počtu her, ale i tak se některé hry drží na žebříčku aktuálně nejhranějších her [7]. Jeho podstatou je umístění dvou týmů do jedné arény, kde bojují proti sobě a snaží se obsadit území nepřátelského týmu. Hra pak zahrnuje různé postavy a role, u kterých je potřeba různých strategických rozhodnutí.

RPG

V *role-playing game* je základ vžití se do samotné postavy, kterou svými rozhodnutími budujete a vylepšujete. Podle postavy se pak dále odvíjí celá hra. RPG obsahuje příběh, mise, interakce s NPC nebo i herní ekonomiku.

Simulace a sport

Jsou to dva úzce spjaté žánry. Cílem simulací je věrně napodobit realitu v počítačovém světě. Hry pak lze využít i jako výukové systémy nebo pro simulaci určitých jevů.

Sportovní hry si ze simulačních her berou mnohé, snaží se přesně zachytit sport, jeho pravidla a zprostředkovat je hráči. Na druhé straně existují hry, které neberou ohled na realitu a nebojí se používat fiktivní prvky.

3.4 Herní zařízení

Postupem vývoje počítačových her přišly různé zařízení. Některé vytvořené primárně pro hráče. Jiné pouze tuto možnost umožňují jako vedlejší funkci. Mezi zástupce, které nejvíce působí na videoherní scéně, se objevují velmi podobné, ale i odlišné zařízení.

3.4.1 Herní automaty

Evoluce her je spjata se vznikem herních automatů. Právě tyto stroje stály za zlatou érou videoher.

Automaty operovaly na základě vhozených peněz, dost podobně jako dnešní hazardní automaty, ale s hazardem neměly nic společného. Automaty stály za vznikem speciálních herních klubů, často přezdívaných *arcade*. Ty byly spíš situované v USA. Zánik těchto heren a automatů má na svědomí technologický posun a možnost domácích herních konzolí. [8]

Nejvíce zmiňované hry, které automaty poskytovaly, jsou *Space Invaders*, *Mario Bros* nebo *Street Fighter*.

3.4.2 Stolní počítače a laptopy

Stále velmi značná část hráčů využívá klasických osobních počítačů. Důvodem může být i velká univerzálnost těchto strojů, které už jsou dnešním standardem v domácnostech, nebo

i technologický pokrok laptopů které, i přes lehkou přenositelnost, jsou schopné se vypořádat bez problémů i s novými herními tituly. [9]

Aktuální trendy stále určuje právě tato platforma. A to díky existenci obrovských herních komunit propojených navzájem. Hráči komunity, spojené s určitou hrou, různě diskutují, vytváří zábavné koláže, sledují živé přenosy, a hlavně spolu hrají. Stále aktuálním tématem je elektronický sport, zkratkou *e-sport*, ve kterém se hráči utkávají proti sobě a poměřují síly v turnajích. I přes aktuální úpadek se stále drží a u některých her je už brán jako součást herní komunity [10].

3.4.3 Mobilní zařízení

Od nástupu pokročilých mobilních technologií došlo k výraznému nárůstu popularity mobilních her. Tyto hry jsou obvykle určeny pro krátkodobé hraní ve volném čase, často na cestách. Navzdory jejich nenáročnému charakteru jsou návykové a zábavné.

Několik let se na nejvyšší příčce držela puzzle hra s názvem *Candy Crush Saga*. Jedná se o hru obsahující tisíce úrovní, ve které je díky propojení se sociálními sítěmi možné vidět postup přátel. Díky výkonovému pokroku se začaly objevovat hry, se kterými se lze setkat u konzolí a počítačů. Jejich funkce a grafika jsou značně omezeny právě pro výkon procesorů v mobilních telefonech [11].

Vývoj mobilních her se technologicky tolik neliší od počítačových her. Nicméně v úvahu se musí brát menší výkon zařízení a ovládání skrz dotykovou obrazovku.

3.4.4 Herní konzole

Funkčně jsou to stále osobní počítače, nicméně jejich hardware a software je přizpůsobený pro lepší výkon. Herní konzole bývají rozdělovány do dvou kategorií:

Stolní konzole

Zpravidla prodávané samostatné zařízení s ovladačem. Obraz pak tedy jde na externí zařízení uživatele, nejčastěji televize. Firmware konzolí je velmi orientovaný pro jednoduché ovládání pomoci gamepadu, ale kromě hraní her obsahuje další funkce jako webový prohlížeč nebo multimediální přehrávač.

Konzole jsou často rozdělovány do generací, přičemž poslední generace je označována jako devátá. Mezi znaky této generace patří velké zvýšení výpočetního výkonu a paměti, připravenost pro virtuální realitu a lepší zapadnutí do trendu *cross-platform* hraní neboli možnosti hraní s uživateli s jiným zařízením. [12] Hlavní představitelé této generace jsou *PlayStation 5* od Sony a Microsoft *Xbox Series*.

Přenosné konzole (Handheld)

Podobně jako stolní konzole obsahují hardware optimalizovaný pro herní výkon, ale navíc obsahují svoji vlastní obrazovku a zpravidla i baterii, zároveň se vyznačují lehčí přenositelností. Mezi hlavní představitele patří ikonický *Gameboy*, *Nintendo DS* nebo *Sony PSP*. Ovšem to

jsou konzole již několik let staré. Pokles zájmu může mít částečně za vinu vývoj chytrých mobilních telefonů. I přes to se zájem o přenosné konzole vrací. Konzole jako *Steam Deck* (Obrázek 1) dokážou, díky nejaktuálnějším technologiím, zvládat velmi náročné nové hry.

Obrázek 1: Steam Deck[1]

3.4.5 Herní příslušenství

Na požitek ze hry může mít podstatný vliv i využití herního příslušenství, které zároveň může hráčům usnadnit ovládání. Hlavním rozdílem od klasických klávesnic je zvětšení přesnosti. Z toho právě nejvíce benefitují hry simulující reálný svět. Při vývoji hry by měla být zvážena i podpora těchto příslušenství, pokud z nich má hra potenciál benefitovat.

Volant s pedály

Herní volanty jsou tvořeny, aby co nejpřesněji kopírovaly pocit a vzhled skutečných volantů z reálných automobilů, proto často obsahují i materiály, které se skutečně používají v automobilovém průmyslu.

Libovolně nastavitelná citlivost, tlačítka nebo i pádlové řazení se dnes berou jako součást běžného vybavení. Dále jsou často součástí pedály obsahující brzdu, plyn a u pokročilejších setů také spojku s řadící pákou.

Největší posun herních volantů přišel s technologií *Force Feedback*. Pomocí speciálních motorů a senzorů lze simulovat vibrace a odpor volantu při zatáčení. Tím je dosaženo mnohem více realistického zážitku. [13]

Joystick

Jedná se o herní zařízení umožňující ovládání her pomocí pohyblivé páky. Díky analogovému vstupu je pohyb přenášen přesně do hry. Dále obsahují i několik tlačítek a vibrační odezvu. Joysticky jsou celkově ergonomické pro pohodlné držení v ruce.

Joysticky nejsou tolik populární, ale jejich přednosti se využívají v leteckých simulátorech nebo závodních hrách.

Gamepad

Zařízení tvarované pro držení v rukou obsahující několik tlačítek, joysticků a analogových páček. Moderní gamepady (Obrázek 2) obsahují i senzory pohybu nebo dotykové plošky pro

[1] https://store.steampowered.com/steamdeck/

lepší ovládaní. Gamepady a jejich prvky používají primárně herní konzole, ale použít se dají i s běžnými počítači nebo mobilními telefony.

Obrázek 2: Xbox Gamepad[2]

3.5 Budoucnost

Pokud vyvíjená hra cílí na úspěch, je potřeba se držet i aktuálních trendů, které se mění každým rokem a novými technologiemi. Součástí trendů je i větší začlenění umělé inteligence do her. To bude zmíněno v kapitole 5.4.

3.5.1 Virtuální a rozšířená realita

Virtuální realita (VR) poskytuje pocit reálné reality uvnitř počítačového prostoru. Načež rozšířená realita (AR) pouze skutečnou realitu rozšiřuje počítačovými informacemi.

Virtuální realita se stává v dnešních dnech více dostupnější. Existují brýle umožňující vstoupit do virtuálního světa. Pro tyto brýle již existují i hry. Součástí brýlí je i zpravidla sada ovladačů do ruky. Ty snímají pohyb a přenáší ho do VR. Alternativou můžou být kamery umístěné v brýlích, které snímají ruce.

Na rozdíl od virtuální reality, rozšířená realita se tolik nevyužívá v herním průmyslu, ale její využití spíše spočívá v předávání počítačově zpracovaných informací. Spíše výjimečného využití v herním průmyslu se našlo v mobilních hrách. Například ve hře *Pokémon GO*, ve které je možnost využít AR k hraní.

Obrázek 3: Rozšířená realita - Pokémon GO[3]

[2]https://www.xbox.com/cs-CZ/accessories/controllers/xbox-wireless-controller
[3]https://niantic.helpshift.com/hc/en/6-pokemon-go/faq/28-catching-pokemon-in-ar-mode-1712012768/

3.5.2 Cloud gaming

Díky technologickému posunu se objevil i *cloud gaming* neboli hraní přes cloud. Jedná se o metodu hraní her, ke které není potřeba výkonné zařízení. Zároveň se jedná o jednu z podmnožin spadajících pod *cloud computing*.

Veškeré výpočty a uložená data jsou umístěny na vzdálených serverech v datacentrech. Tudíž není potřeba stahování her. Jediné, co je potřeba k této technologii, je velmi stabilní a rychlý internet, přes který se posílá obraz ze serveru klientovi. Klient poté odesílá zpět na server informace o vstupech (ovládání).

Výhody jsou ve vstupních nákladech. Není potřeba pořizovat herní počítač nebo konzoli, ale stačí pouhé předplacení služby, kvalitní internet a zobrazovací zařízení. Tím může být televize, počítač nebo mobilní telefon. S kvalitním internetem přijde i zásadní nevýhoda. Cloud gaming není vhodný například pro cestování, při kterém připojení může kolísat. [14]

Na scéně účinkuje pár hlavních představitelů, které poskytují cloud gaming služby. Hlavním představitelem je Geforce Now od NVidia. Díky předplatnému je možné zahrát si hry, které uživatel vlastní. Nabízí i omezenou verzi pro vyzkoušení. Druhým hlavním představitelem je Xbox Cloud Gaming, který je zdarma v rámci Xbox Game Pass předplatného, ve kterém předplatitel získá zdarma i přístup do velké knihovny her.

3.6 Postup tvorby počítačových her

Prvním krokem každé tvorby nové počítačové hry je nápad a návrh. Návrh by měl být důkladně promyšlený. S tím souvisí nastavení jednotlivých cílů, způsobů zaujmutí publika, a i vyhledání možné konkurence. V této fázi zpravidla vzniká GDD (*Game Design Document*), který obsahuje veškeré informace, jak by hra měla vypadat. Zároveň napomáhá k organizaci práce nebo k marketingu připravované hry. [15]

Základní technický prvek každé počítačové hry je herní engine. Ten má na starost kořenové funkce videoher, jako je vstup od uživatele, vykreslování, zvuk nebo definici objektů. Při programování nové hry je k dispozici možnost vytvoření vlastního enginu. Tím se dosáhne plné kontroly nad ním a může být přizpůsobován podle potřeb vytvářené hry. Druhou možností je využití enginu třetí strany.

Po obstarání herního engine nastává další fáze. Zpravidla se začíná s modelováním a texturováním terénu, scén, postav a objektů. S tím mohou pomoci knihovny s 3D objekty pro volné použití. Následuje i animace objektů. Nutnost je také zakomponování fyziky a kolizí objektů, pokud již tyto problémy nejsou řešeny v samotném enginu.

Zároveň s tvorbou samotných objektů se přidává i programování logiky hry. Logika souvisí se žánrem hry, podle které se vývoj odvíjí. Tvoří se zde různé základní prvky jako zdraví hrdiny, techniky reálného světa, herní ekonomika, interakce s objekty a mnohem více. Správné vyvážení ekonomiky nebo přenesení řízení a chování vozidla vyžaduje spoustu času na vývoj.

Pro splnění charakteristiky hry je nutno přidat cíle, motivační kontrolní body a celkový příběh, pokud je na něm hra postavena. S tím souvisí i tvorba *cutscene*, krátkých vyprávěcích vložek, interakce s jednotlivými NPC a samotná tvorba hlavní postavy.

Hra se může považovat za hotovou v případě, kdy je provedeno dostatek testování a odladění. Při větších projektech se vyplatí hry testovat ve větších skupinách, případně vydat testovací verze pro užší skupinu hráčů. Výsledek testování přinese množství chyb a jejich možný způsob odstranění, nebo i požadavky na lepší optimalizaci. Odladění hry může zasahovat přes všechny vrstvy až do samotného enginu. [15]

Pro udržení hry je nadále potřeba přidávat nový obsah, opravovat nově nalezené chyby a optimalizovat hru pro nové technologie. Nedílnou součástí je i marketing postavený kolem hry.

4 Herní enginy

Jedná se o základní stavební jádro počítačové hry obsahující soubor modulů, které nespecifikují logiku nebo prostředí hry. Naopak zahrnuje moduly pro zpracování různých základních funkcí her.

Engine může být vyvíjen společně s počítačovou hrou. Častokrát je takový engine spjatý pouze s určitou hrou a je vytvořen pouze pro ni. Běžnou praxí je také vývoj enginu v rámci vývojového studia, který se pak používá pro více her, ale není publikován samostatně. [16]

Herní engine se dá považovat i jako *framework*. Zde se jedná o již vytvořené enginy, které lze využívat k vývoji. Podporují větší množství funkcí, aby pokryly větší rozsah možností tvorby nebo je na nich založeno celé vývojové prostředí.

4.1 Funkce

Jednotlivé enginy se mohou velmi lišit ve svých funkcích a stylu implementace, ale obecně se dá říct, že mají na starost následující technické prvky počítačové hry.

Vykreslování

Obsahuje několik kroků jejichž výsledkem je výstup na obrazovku uživatele. V případě 3D hry se celková scéna, složená z vertex bodů, přepočítává do pohledu uživatele a ořezává velikostí obrazu. Dále do vykreslování patří i samotné osvětlení ve scéně, ořezávání neviditelných částí a finální rasterizace.

Jednotlivé enginy fungují na odlišných implementacích. Zároveň můžou umožňovat přidání nových funkcí nebo samotnou úpravu zobrazovací pipeline.

Objekty a kolize

Objekty jsou úzce spjaté se samotným vykreslováním. Jejich správu a reprezentaci musí engine také podporovat. Vytváří se 3D modely herních objektů pomocí polygonů, vertexů a textur.

Fyzika

Fyzika dodává hře určitý stupeň reality. Přidává do hry gravitace objektů anebo jejich reakce na kolizi. Herní fyzika je volena podle žánru, například u simulací je dbáno, aby co nejvíce odpovídaly realitě.

Vstup od uživatele

Engine musí mít zakomponované předávání vstupů od uživatele. Řeší se zde zařízení, které používá, jednotlivé klávesy, poloha kurzoru nebo míra vstupu od pokročilejších zařízení. Jak moc je pedál u volantu stlačen, nebo jak moc je posunutý joystick a jakým směrem.

Zvuk

Zvuk je podstatná část počítačových her. Přidává další zpětnou vazbu pro hráče, zpříjemňuje jeho zážitek a celkově dodává hře svoji atmosféru. Engine má na starost okolní zvuky světa, rozhovory, různé efekty a hlasitost podle vzdálenosti od objektu.

4.2 Komerční enginy

Jedná se o již existující herní enginy, které lze použít pro vývoj počítačových her. Na těchto herních enginech mohou být dokonce postavena celá vývojová prostředí, která usnadňují vývoj. Kolem těchto enginů existují také komunity, které vytvářejí další obsah a knihovny nebo poskytují výukové návody pro vývoj. Díky tomu je pro začátečníky vývoj značně ulehčen.

Díky použití komerčních enginů lze vývoj počítačových her výrazně urychlit. Navíc lze snížit náklady na vývoj. Používání takových enginů však může být náročnější na naučení nebo může být omezující v přizpůsobení samotného enginu.

4.2.1 Unity

Herní engine Unity je považován za jeden z nejpoužívanějších herních enginů v dnešní době. Byl vydán společností *Unity Technologies* v roce 2005 za účelem otevření více možností pro vývoj veřejnosti. I přes stáří, je engine pravidelně aktualizován a obsahuje nejnovější technologie pro vývoj her.

Unity poskytuje licence, v rámci kterých je pro studenty a individuální uživatele používání zdarma. Pro tyto individuální uživatele platí pravidlo maximálního zisku 100 tisíc dolarů za jeden rok. V případě překročení limitu je nucen si zaplatit plán *Unity Pro*. Tím získá výhody typu lepší zákaznické podpory nebo přístup ke cloudovému uložišti. V případě využití větším vývojářským týmem jsou k dispozici plány *Unity Enterprise* a *Unity Industry* poskytující ještě větší množství výhod. [17]

S využitím Unity vzniklo obrovské množství her, zejména pak žánru indie. Mezi nejznámější hry postavené na tomto enginu patří mobilní hra *Pokémon GO*, pro počítače *Subnautica* nebo *Fall Guys*.

4.2.2 CryEngine

Jedná se o herní engine podporující vývoj her na většině populárních zařízení. Stojí za ním německé vývojové studio Crytek.

I tento engine je postaven na royality systému. Pokud vytvořená hra přesáhne 5 tisíc dolarů ročního příjmu, je vývojář povinen odvádět 5 % ze svých hrubých příjmů. [18]

CryEngine získal svoji reputaci díky úspěšnosti v napodobení reality a jistě také díky otevřenému zdrojovému kódu samotného enginu v programovacím jazyce C++. Obsahuje i vlastní obchod s objekty.

Mezi známé herní tituly, které jsou postaveny na CryEnginu, patří *Far Cry* nebo *Crysis*.

4.2.3 Unreal Engine

Unreal Engine se nedá považovat jenom za herní engine. Jak je uvedeno na webových stránkách, jedná se o „*kompletní sadu nástrojů pro tvorbu her, architektonické a automobilové vizualizace, tvorbu lineárního filmového a televizního obsahu, vysílání a produkci živých akcí, školení, simulace a další využití v real-time aplikacích.*“ [19]

První použití Unreal Enginu proběhlo v roce 1998, a to s vydáním hry *Unreal*. Za enginem stojí americká firma Epic Games. Od té doby se postupně z enginu určeného pro střílečku stala již zmiňovaná sada nástrojů pro vývojáře a filmaře. Základní balíček s enginem je volně dostupný ke stažení. Zároveň má velmi podobné licence jako Unity. Unreal Engine je zdarma do doby, než projekt vydělá méně než 1 milion dolarů celkem. Po přesáhnutí začíná platit poplatek 5 % z tržeb. Druhou možností jsou individuální licence pro podniky, které obsahují benefity, anebo menší poplatky z tržeb.

Blueprints

Celým názvem *Blueprint Visual Scripting* je systém grafického rozhraní ke skriptování v Unreal Engine. Poprvé se objevil ve verzi 3 pod jménem Kismet, ve verzi 4 došlo k přejmenování a úpravě grafického rozhraní.

Podobně jako jiné skriptovací jazyky slouží k definici objektově orientovaných tříd a objektů. Díky grafickému rozhraní umožní využívat celou řadu konceptů a nástrojů, které jsou běžně přístupné pouze programátorům. Tento systém je vhodný i pro méně zkušené programátory, kteří mají alespoň základní znalosti algoritmizace. Zároveň všechny blueprints jsou implementované v jazyce C++ a umožňují vzniklé systémy nadále rozšiřovat [20].

4.2.4 Porovnání Unity a Unreal Engine

Tito dva zástupci herních enginů jsou považováni za nejoblíbenější pro malá studia a jednotlivce vyvíjející hry.

Unity nabízí solidní základy pro začátečníky a bohatou dokumentaci. Engine velmi univerzální pro 2D, 3D, mobilní hry a širokou škálu žánrů. Jednou z jeho nevýhod je méně realistická grafika ve srovnání s Unreal Enginem.

Oproti tomu Unreal Engine nabízí systém blueprints, který je velmi intuitivní i pro jednotlivce, kteří neovládají programování na vysoké úrovni. Grafická stránka je navíc velmi pokročilá a realistická. Unreal Engine také zjednodušuje vývoj pro virtuální a rozšířenou realitu. Mezi jeho slabiny patří horší vývoj pro 2D a mobilní hry. Kromě toho jsou systémové požadavky na počítač vyšší než u Unity [21].

5 Umělá inteligence

Definovat umělou inteligenci není snadné. Ve skutečnosti neexistuje žádná obecně přijímaná definice umělé inteligence. Proto bude probráno několik možných definic pro co nejpřesnější představení umělé inteligence.

V širokém pojetí je umělá inteligence ztotožňována s algoritmy. To však není příliš vypovídající definice. Pojem algoritmus označuje konkrétní instrukci pro řešení problémů nebo provedení výpočtů. Tedy v tomto případě by měla umělá inteligence zahrnovat operace jako kapesní kalkulačky. [22]

V užším pojetí se umělá inteligence snaží o napodobování lidské inteligence počítači. Ani zde se nejedná o přesnou definici. V dalších kapitolách budou představeny systémy, které se pouze tváří jako umělá inteligence, ale jinak jsou to celkem jednoduché programy.

Přesnější definice byla vytvořena skupinou odborníků Evropské komise. V překladu definice představuje umělou inteligenci jako systémy, které inteligentním chováním analyzují své prostředí a přijímají opatření, s určitým stupněm autonomie, k dosažení specifických cílů. I tato definice obsahuje několik mezer, které nejsou naprosto jasné. Nelze definovat, o jaký určitý stupeň autonomie se jedná. [23]

Jak už bylo řečeno, neexistuje obecně přijímaná definice. Problém definovat umělou inteligenci zasahuje i do definice lidské inteligence. I ta je velmi problémová definovat. [22]

5.1 Slabá a silná AI

V definici napomáhá rozdělení na slabou (*weak* nebo *narrow*) a silnou (*strong*) umělou inteligenci. Silná umělá inteligence je v současné době ve světě pouze hypotetická. Pojednává o umělé inteligenci, která dokáže přemýšlet jako člověk a je si vědoma sama sebe. Měla by být schopna řešit problémy a učit se v plné samostatnosti. Vytvoření takové umělé inteligence se zatím nepodařilo dosáhnout.[24]

Na druhou stranu je slabá umělá inteligence již široce využívána k řešení velkého množství problémů. Označují se tak systémy naprogramované k řešení konkrétní úlohy. Rozlišuje se ještě na další druhy, a to podle toho, zda se učí a zlepšují tím svůj výkon, nebo pracují na základě stanovených pravidel.

5.2 Turingův test

Již v raných začátcích umělé inteligence přišla otázka, jak rozpoznat, zda daný program myslí jako člověk. V roce 1950 byl uveřejněn test Alanem Turingem, který by tuto otázku měl řešit. Základem tohoto testu je imitační hra.

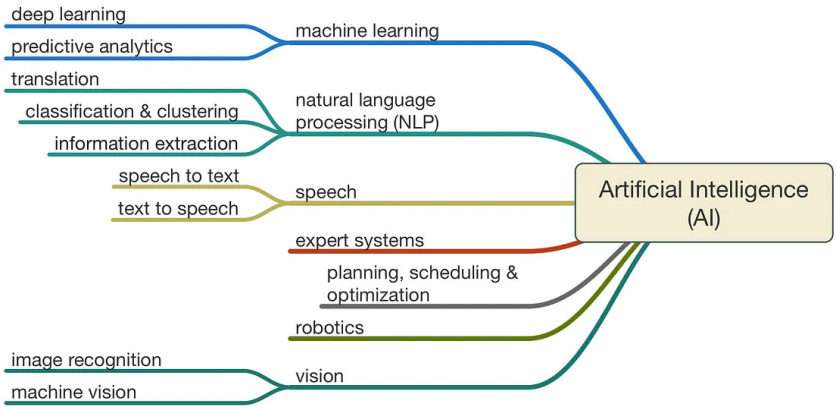
Podstatou je, aby člověk rozhodl na základě komunikace, bez fyzického kontaktu, zda komunikuje s člověkem nebo se systémem s umělou inteligencí. V případě, pokud není člověk schopen rozlišit odpovědi systému od odpovědí člověka, je systém označen jako inteligentní. Rozšířený Turingův test navíc zahrnuje fyzickou interakci, a to využitím počítačového vidění a robotiky.

Test však čelí mnoha kritikám. Jednou z nejznámějších je argument čínského pokoje. Argument pojednává o osobě, která neumí čínsky a je zavřená v pokoji. Osobě je dán soubor pravidel pro manipulaci s čínskými symboly. Dále postupně obdržuje čínské symboly a na základě pravidel s nimi manipuluje a odesílá zpátky. Tudíž pro osoby vně pokoje se zdá, že osoba v pokoji umí čínsky. John Searle, který navrhl tento argument, pojednává, že stroj, který projde Turingovým testem, nemusí být inteligentní, ale pouze manipuluje se symboly podle pravidel. [25]

5.3 Obory umělé inteligence

Umělá inteligence se definuje i jako vědecký obor informatiky, který se zabývá tvorbou inteligentních systémů, které se dokáží učit a jednat autonomně.

Obory umělé inteligence se od počátku rozdělují do různých odvětví (Obrázek 4). Každé odvětví se zaměřuje na jinou technologii nebo na jiná řešení problémů. Mezi hlavní představitele odvětví patří strojové učení, zpracování jazyka, expertní systémy nebo počítačové vidění. Jednotlivé odvětví se dále rozděluje na podkategorie a zároveň postupem času určitě přibudou další.



Obrázek 4: Obory umělé inteligence[4]

5.3.1 Neuronová síť

Umělé neuronové sítě jsou programy, které spadají do kategorie strojového učení. Jejich funkce je podobná funkci lidského mozku díky procesům, které napodobují spojení mozkových neuronů. Každá neuronová síť se skládá z vrstev uzlů, označovaných jako umělé neurony. Obsahují vstupní, skryté a výstupní vrstvy. Každý uzel se připojuje k ostatním a má přiřazenou vlastní váhu a prahovou hodnotu. Pokud je výstup některého jednotlivého uzlu vyšší než zadaná prahová hodnota, tento uzel se aktivuje a odešle data do další vrstvy sítě. V opačném případě nejsou další vrstvě sítě data předána. [26]

[4]https://medium.com/ml-ai-study-group/ai-mind-map-a70dafcf5a48

Typy

Neuronové sítě lze rozdělit na různé typy, které se používají k různým účelům. Níže uvedené typy reprezentují nejběžnější typy neuronových sítí, se kterými je možnost se setkat v běžných případech použití.

Nejběžnějším typem sítí jsou *Feedforward* neboli vícevrstvé perceptrony (MLP). Skládají se ze vstupní vrstvy, skryté vrstvy nebo vrstev a výstupní vrstvy. Ačkoli se označují jako sítě s perceptrony, ve skutečnosti se skládají ze sigmoidních neuronů. Důvodem je většina problémů, které jsou nelineárního typu. Obvykle se tyto sítě trénují vkládáním dat. Jsou základem pro další druhy neuronových sítí. [27]

Konvoluční neuronové sítě (CNN) jsou velmi podobné typu *Feedforward* síti. Využívají se primárně pro rozpoznávání obrazu a vzorů, nebo i pro počítačové vidění. Sítě využívají principy lineární algebry. Pro příklad násobení matic pro identifikaci vzoru v obraze. [28]

Neurony rekurentní neuronové sítě (RNN) jsou rozpoznatelné podle zpětnovazebních signálů. Tyto algoritmy se používají především při použití časových dat k předpovědím budoucích výsledků. Příkladem jsou předpovědi akciového trhu nebo prodeje produktů. [29]

Fungování

Po určení vstupní vrstvy se jí přiřadí váhy. Tyto váhy pomáhají určit důležitost jednotlivých proměnných, přičemž větší váhy přispívají k výstupu významněji než ostatní vstupy. Všechny vstupy se poté vynásobí příslušnými váhami a sečtou. Výstup pak prochází aktivační funkcí, která určuje výstup. Pokud tento výstup překročí určitou mez, uzel se aktivuje a data se předají do další vrstvy sítě. Výsledkem je, že výstup jednoho uzlu se stává vstupem dalšího. Tento proces předávání dat z jedné vrstvy do druhé definuje tuto neuronovou síť jako dopřednou síť neboli *Feedforward*. [26]

Neuronové sítě využívají sigmoidní neurony, které se vyznačují tím, že mají hodnoty mezi 0 a 1. Vzhledem k tomu, že se neuronové sítě chovají podobně jako rozhodovací stromy a kaskádovitě přenášejí data z jednoho uzlu do druhého, snižuje se vliv jakékoli změny jedné proměnné na výstup kteréhokoli uzlu a následně na výstup neuronové sítě.

Během trénování modelu je nutné vyhodnotit jeho přesnost pomocí ztrátové funkce. Tato funkce se běžně označuje jako střední kvadratická chyba (MSE). V konečném důsledku je cílem minimalizovat nákladovou funkci, aby se zajistilo správné přizpůsobení pro každé pozorování. Model upravuje své váhy s využitím nákladové funkce a *reinforcement* učení, aby dosáhl konvergence známé také jako lokální minimum. Proces, kterým algoritmus upravuje své váhy, probíhá prostřednictvím gradientního sestupu, což modelu umožňuje určit směr, kterým by měl postupovat, aby snížil chyby nebo minimalizoval nákladovou funkci. S každým trénovacím příkladem se parametry modelu upravují tak, aby postupně konvergovaly k minimu. [26]

Výhody a využití

Sítě spoléhají na tréninkových datech, se kterými se učí a vylepšují svojí přesnost v průběhu času. V případě, že je jejich trénink správně navržený, stanou se velmi silnými nástroji v informatice. Jeden z nejznámějších příkladů neuronové sítě je vyhledávač Google.

Nejčastěji se neuronové sítě využívají k rozpoznávání obrazu, zpracování přirozeného jazyka, předvídání budoucích událostí a generování obsahu.

Schopnost učení z dat a zlepšování v průběhu času, řešení komplexních problémů nebo zpracovávání velkých objemů dat. To jsou značné výhody neuronových sítí. I přes to můžou být náročné na výpočet nebo hůře interpretované. [30]

5.3.2 Genetické algoritmy

Jedná se o obor umělé inteligence, který je inspirovaný principem biologické evoluce. Využívají se k řešení složitých optimalizačních úloh, u kterých nelze implementovat jednotný algoritmus.

Genetické algoritmy pracují s populací jedinců označované jako generace. Každý jedinec představuje jedno možné řešení daného problému. Jedinec v sobě uchovává řetězec, který zobrazuje postup řešení problému. [31]

Jedinec nadále obsahuje svoji hodnotu označovanou jako *fitness*, která označuje kvalitu jeho řešení. Hodnota se počítá specificky podle daného problému.

Na základě této hodnoty se využívají operátory, které upravují řetězce jednotlivců, ze kterých pak plyne nová generace.

Operátory

Prvním operátorem je selekce. Na základě úspěšnosti jednotlivých jedinců k poměru úspěšnosti celé generace jsou vybráni ti, kteří jsou blíže k řešení. Následně tito vybraní jedinci utvoří přechodovou populaci, na kterou budou aplikovány operátory křížení a mutace.

Z přechodové generace jedinci poskytnou část svých informací a křížením těchto informací vzniká nová populace. Přesněji se zvolí dva jedinci, u kterých jsou jejich řetězce informací náhodně rozděleny. Následně jsou vytvořeny nové populace zkřížením řetězců. [32]

Operátor mutace upravuje atributy řetězce s určitou odchylkou, tím vznikají nové vlastnosti, které se podobají těm předchozím. [31]

Výhody a využití

Mezi výhody genetických algoritmů patří jejich univerzálnost nasazení a řešení široké škály problémů. Dobře implementované algoritmy jsou robustní a odolné vůči šumu. Naproti tomu můžou být výpočetně náročné v případě řešení složitých problémů. Problémem může být i hledání vhodné funkce pro výpočet *fitness* u jedinců.

Genetické algoritmy mají široké možnosti využití. Využívají se v trénování neuronových sítí, optimalizaci výrobních procesů, bioinformatice, financích a mnoho dalšího.

5.4 AI fungující na základě pravidel

Tyto systémy jsou většinou mnohem jednodušší a operují na základě větvení pravidel. Nemají schopnost porozumět datům nebo se rozhodovat. Vše spoléhá na předem připravených pravidlech.

Často se může označovat jako falešná umělá inteligence. Jedná se o implementace systémů, které mají představovat vyšší formu inteligence, ale pro jejich účel není nutné využití komplexnějších systémů. Zpravidla mají za úkol, aby si uživatel pouze myslel, že komunikuje s člověkem nebo propracovanější umělou inteligencí.

5.4.1 Fungování

Systémy jsou tvořeny na *Ruled-based programming* neboli programování balíčku pravidel a rozhodovacích stromů. Tím přesně určují, v jakých situacích se mají, jak zachovat. Všechna tato pravidla jsou vytvořena člověkem, tudíž si nezakládají na žádném učícím se procesu.

Vstupy do takových systémů musejí být specifické bez jakékoliv odchylky. Pro každý vstup je vytvořeno právě jedno pravidlo. A to z důvodu, že systém nemůže rozpoznat a porozumět datům, které obdržel.

Díky předem definovaným pravidlům mají tyto systémy limitované možnosti a dokážou provádět pouze specifické operace na základě pravidel. Nemají žádnou schopnost adaptovat se novým situacím.

Použití

I přes omezené možnosti se systémy hodí do určitých situací, a to primárně díky jednodušší implementaci. Dnes spousta společností nabízí virtuální asistenty na svých webových stránkách. Tito asistenti pomáhají zákazníkům řešit základní problémy nebo je selektovat pro budoucí zpracování. Naleznout takové systémy je možné na internetových obchodech nebo stránkách poskytovatele internetu.

5.4.2 Počítačové hry

S falešnou umělou inteligencí se lze setkat i v počítačových hrách. Současné NPC fungují na základě stanovených pravidlech a jejich chování zůstává stejné. Jedná se o nedílnou součást dnešních her, ať už se jedná o ovládání nepřátel nebo chování NPC ve světě.

S rozšířením slabé umělé inteligence lze ale předpokládat, že některé systémy, založené na pravidlech, budou nahrazeny učící se umělou inteligencí. Tím se dosáhne mnohem více propracovanějších her a postav. Jejich odpovědi můžou být přesnější a chytřejší. Reakce mohou být mnohem propracovanější směrem k hernímu světu.

Použití bude i v rámci vývoje her. V prezentaci budoucí aktualizace 5.4 pro Unreal Engine bylo představeno procedurální generování map pomocí umělé inteligence.[5] Případně je lze využít i ke zlepšení grafické stránky her. *Upscalling* existuje už delší dobu. Používá se ke zvětšení rozlišení obrazu při snaze o zachování co nejlepší kvality. Bývá používán na slabších

[5] State of Unreal 2024: `https://www.youtube.com/watch?v=yVoQ5AH_wxI`

zařízeních, ve kterých bývá hra vykreslená na menší rozlišení a poté je obraz přeškálován na rozlišení výstupního zařízení. [33]

5.5 Knihovny pro strojové učení

Knihovny vznikají za účelem zjednodušení vývoje software a i pro strojové učení již takové knihovny existují a jsou velmi často využívané. Velké množství funkcí je již implementováno a při jejich používání lze nimi obalit vytvořený program.

5.5.1 TensorFlow

Jedná se o knihovnu stvořenou společností Google v roce 2015. Zároveň je to nejvíce populární framework v dnešní době. Využívají ho nejen velké společnosti, ale i datoví vědci a menší vývojáři.

TensorFlow je určen primárně pro programovací jazyk Python, ale již byl převeden i do jiných jazyků jako Java nebo C++. [34]

5.5.2 PyTorch

Hlavní oponent TensorFlow je PyTorch. Byl vytvořen vytvořen společností Facebook v roce 2017 a nyní ho spravuje Meta AI. Je založen na knihovně Torch. Nabízí jednodušší implementování a trénování neuronových sítí. Komunita kolem PyTorch je velmi aktivní a každým dnem se publikují vytrénované modely ke stažení. [35]

PyTorch funguje pouze s programovacím jazykem Python, což dost oslabuje jeho pozici oproti TensorFlow.

5.5.3 Keras

Framework Keras je postavený na TensorFlow. Je méně populární než již zmiňované knihovny. Oproti tomu je přívětivější pro začátečníky.

Díky Keras lze vytvářet komplexní modely jen pomocí pár řádků kódu. Zároveň ale nenabízí tolik možností konfigurace. [36]

6 Návrh AI

Využití umělé inteligence k řešení her lze ve velkém množství druhů a žánrů. Pro tuto práci byla vybrána závodní hra. Umělá inteligence tedy bude zodpovědná za ovládání závodního vozu na trati a měla by se snažit o docílení co nejlepšího času.

6.1 Příprava počítačové hry

Pro tvorbu počítačové hry bylo zamýšleno využití frameworků nebo herních enginů třetích stran. Využití *OpenGL* nebo *PyGame* by přineslo spoustu výhod jako menší velikost projektu nebo celková upravitelnost oproti herním enginům. Ale zásadní nevýhodou by byla složitost a trvání celkové tvorby počítačové hry. I proto byl pro vývoj zvolen herní engine Unreal Engine.

Unreal Engine obsahuje zároveň i šablonu pro vývoj závodních her. Šablona obsahuje trať a naprogramovaný závodní automobil, včetně veškeré logiky zatáčení a kolizí. Další výhodou je možné prozkoumání již vytvořených pluginů pro implementaci strojového učení. Případně může být vytvoření umělé inteligence nápomocné dalším vývojářům.

Pro stažení Unreal Engine[6] je nutné si stáhnout a nainstalovat Epic Games Launcher. Tento program slouží jako obchod pro počítačové hry a jejich správu (instalace, spouštění), vedle toho obsahuje i možnost stažení Unreal Engine a k tomu Unreal Engine Marketplace, ve kterém se dají kupovat a stahovat přídavné pluginy a balíčky s 3D modely. Po spuštění Unreal Engine stačilo pouze založit nový projekt ze šablony pro závodní hry.

6.2 Návrh implementace

Myšlenkou bylo vytvořit jednoduchou neuronovou síť pouze v Unreal Enginu, která bude řešit zmiňovanou závodní hru.

První fází implementace byl samotný návrh a zprovoznění umělé inteligence. Díky předem vytvořené šabloně pro závodní hry v Unreal Enginu je k dispozici model auta, jeho základní prvky a ovládání jsou již implementovány. Na tyto prvky se navázaly jednotlivé funkce pro zprovoznění neuronové sítě.

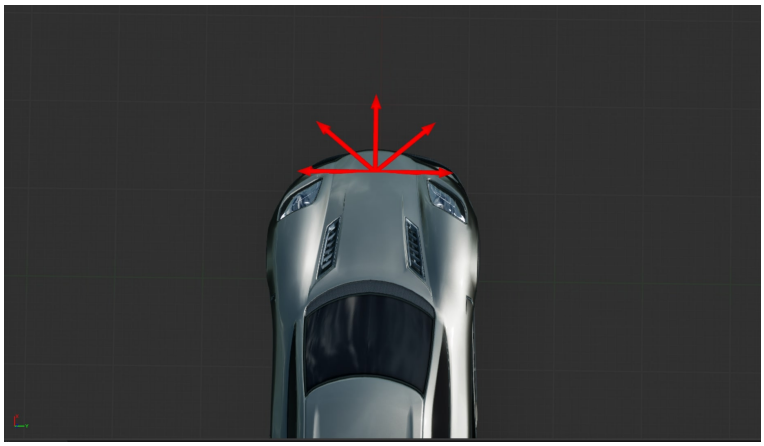
Podrobnější popis fungování

Auto obsahuje senzory vzdálenosti od objektů. Tyto vzdálenosti jsou vstupní hodnoty do neuronové sítě. Na základě predikce výstupy znázorňují, jak auto má zatáčet. Vzhledem k senzorům bylo potřeba vložit model trati obsahující zdi. Trénování neuronové sítě probíhá skrz genetické algoritmy. Z vytvořené generace se vybírají nejlepší jedinci, kteří zajely trať nejlépe. Z vybraných jedinců se poté utvoří nová generace.

[6] `https://www.unrealengine.com/en-US/download`

6.3 Senzory vzdálenosti

Jak bylo zmíněno v úvodu, vytvářené auto musí obsahovat nějakou formu senzorů, které budou zjišťovat aktuální vzdálenosti od objektů. Pro tyto výpočty se přidalo do kostry auta pět prvků *Arrow* (Obrázek 5). Ty umožní skrz funkci *Line Trace* vypočítat vzdálenosti od šipky do první kolize. Těchto pět šipek bylo připevněno v různých úhlech, přesněji v -90°, -45°, 0°, 45° a 90°. Pro výpočet vzdálenosti byla vytvořena funkce *LineTrace*, která na vstupu bere



Obrázek 5: Přidané senzory[7]

samotný objekt *Arrow*. V průběhu funkce získá lokaci šipky ve světě a jeho forward vektor. Následuje funkce *Line Trace by Channel*, do které vstupují dvě hodnoty *Start* a *End*. Do začátku se vloží pozice ve světě a do konce vynásobený vektor sečtený s pozicí ve světě. Poté lze z funkce vytáhnout samotnou vzdálenost, která je obsažena ve výstupu. V automobilu se vytvoří událost *GetData*, která zavolá tuto funkci pro všechny šipky a uloží je do proměnných.

6.4 Závodní trať

K modelování závodní dráhy byl použit nástroj *Spline*, který nabízí možnost vygenerovat křivku, na kterou lze připojit 3D model a následně jej na základě křivky vytvarovat a vyplnit. Model závodní dráhy byl vytvořen pomocí programu *Blender*[8] a připomíná tvar písmene U. Stěny jsou zde nezbytné pro snímání vzdálenosti vozidla. Kromě toho bylo potřeba i zkontrolovat, zda jsou kolize správně nastaveny.

6.5 Datové struktury

Bylo potřeba nadefinovat základní stavební prvky neuronové sítě. Jedná se o datové struktury, které budou využity při tvorbě neuronové sítě a genetických algoritmů.

AI Weight Structure

Jedná se o třídu, která drží v sobě rozhodovací váhy všech neuronů. Nadefinováno je zde 15 hodnot typu *float*. Tyto hodnoty napomáhají k rozhodnutí o výsledku.

[7] Zdroj: vlastní

[8] `https://www.blender.org/`

AI Structure

Drží v sobě data neuronové sítě neboli jedince v generaci. Obsahuje v sobě třídu *AI Weight Structure* a celočíselnou hodnotu *Score*, která je důležitá pro genetický algoritmus.

Best AI List

Třída pro mezigenerační uchovávání třech nejlepších jedinců vybraných z předchozí generace. Obsahuje tedy tři hodnoty typu *AI Structure*.

6.6 Neuronová síť

Před tvorbou neuronové sítě bylo potřeba definovat co je *Neuron*. V programu představuje funkci, která přijímá celkem 5 vstupů a 5 váhových hodnot. Uvnitř funkce se vynásobí vstupy se svými váhami, poté se výsledky sečtou do jednoho čísla. Po vydělení je číslo posláno do další nové funkce *Sigmoid*, která vrátí pravděpodobnost zatáčení vozidla.

Funkce *Sigmoid* je pouhou implementací matematické funkce (Vzorec 1), která slouží pro transformaci hodnot do intervalu (0,1).

$$f(x) = \frac{1}{1 + e^{-x}} \tag{1}$$

Implementace

Celá síť je implementována do třídy automobilu, která byla již obsažena v šabloně. Ve třídě je vytvořena událost *Neural Network*, která v prvním kroku zavolá *GetData* pro získání vzdáleností.

Následuje vložení funkce *Neuron*, a to celkem třikrát. Do vstupů neuronů jdou hodnoty získané skrz *GetData* a aktuální váhové hodnoty. Výsledek funkcí je zpracovaný do pole, z kterého je zvolena maximální hodnota, která se převede na datový typ *integer*. Následně je výsledek nastaven do řízení auta. Nakonec tato událost bude volána každý *event tick*. Co se týče nastavování plynu, to řešené není a jeho řešení bylo plánováno po testování funkčnosti samotného zatáčení. Proto byl plyn nastaven staticky na určitou hodnotu.

6.7 Genetický algoritmus

Po vytvoření neuronové sítě bylo načase implementovat i genetické algoritmy, které mají na starost trénování a vylepšování umělé inteligence.

Princip spočívá ve vytvoření počáteční generace automobilů, přičemž jejich váhové hodnoty jsou přiřazeny náhodně. Každý jedinec začíná s nulovým skóre. Do každého uzlu dráhy *Spline* je vložen kontrolní bod, jako motivační cíl, který zvyšuje skóre jedince. Čím více kontrolních bodů jedinec projede, tím vyšší je jeho skóre. Po uplynutí předem stanoveného času jsou z kompletní generace vybráni tři jedinci, kteří dosáhly nejvyššího skóre. Ti jsou přidáni do dříve vytvořeného seznamu *Best AI List*.

Následující generace se vytváří na základě předchozích nejlepších jedinců. Hodnoty vah se vynásobí malými odchylkami a poté se nastaví jako nové hodnoty pro jedince další generace.

Další vytvořenou třídou je *Training Manager*. Úkolem této třídy je tvořit jednotlivé generace, vkládat je do herního levelu, vybírat tři nejlepší jedince a zobrazovat zprávy o aktuální generaci, která zrovna projíždí trať.

6.8 Testování a výsledek

První testování tohoto řešení je v celku úspěšné. Projíždění trasy se s každou generací zlepšuje. V případě další implementace funkcí a odladňování má tento model potenciál konkurovat člověku. Bohužel se postupem času objevilo několik vad tohoto návrhu.

Zjištěné vady návrhu

Prvním zjištěným nedostatkem, v úvodu testování, byl výkon. Editor byl při trénování zpomalený a v několika pokusech Unreal Engine ohlásil nedostatek paměti grafické karty. Během dalšího testování byl objeven druhý problém se systémem kontrolních bodů. Některá auta se otáčela a začala získávat skóre jízdou v opačném směru trati. Pokud se takový jedinec dostane mezi tři nejlepší, zmate to většinu následujících generací.

Řešení uvedených problémů je možné. Problém s výkonem lze vyřešit snížením počtu jedinců v generaci, což by výrazně uvolnilo paměť, ale zároveň zpomalilo proces trénování. Systém kontrolních bodů by měl být přepracován a ošetřen od nežádoucích situací. Ideální implementací by bylo vložení kolizního boxu, který by se v průběhu času posouval podél křivky. V případě kolize by byl jedinec vyřazen, než by zůstaly pouze poslední tři jedinci.

Po celkovém prozkoumání však bylo zjištěno, že tento návrh neumožňuje pokročilejší učení umělé inteligence. V případě nasazení vytrénovaného jedince na jinou trasu by ji, vzhledem k pevnému nastavení vah neuronové sítě, nebyl schopen projet. Po opravě chyb by byl model použitelný, ale hlavním cílem je vytvořit propracovanější umělou inteligenci schopnou zvládat různé tratě. Proto je potřeba hledat alternativní řešení.

7 Druhý návrh

Jedná se o rozšíření původního návrhu a náhradu samotné neuronové sítě. Myšlenkou je zde zapojit samotnou neuronovou síť skrz framework pro umělou inteligenci. Taková neuronová síť by přijímala data ze senzorů a vracela by údaje týkající se řízení a zrychlení auta.

V Unreal Engine Marketplace je sice k dispozici knihovna pro tvorbu neuronových sítí, ale není zdarma. Proto by bylo lepší použít některou z volně dostupných variant frameworků. Tyto frameworky však nejsou podporovány Unreal Enginem. Engine má vestavěný jazyk Python, který by bylo možné pro tyto účely použít. Nejpoužívanější frameworky, které jsou představené v kapitole 5, jsou přímo implementovatelné pro Python. Po prozkoumání této možnosti se dospělo k závěru, že toto řešení není proveditelné. Vestavěný Python lze použít pouze pro vývojové řešení a nelze jej použít během spuštěné hry.

7.1 Řešení komunikace

Vzhledem k výše zmíněné nepoužitelnosti prostředí Python v samotném Unreal Engine bylo třeba najít řešení pro komunikaci mezi Unreal Engine a programem s neuronovou sítí. V obchodu Unreal Engine je k dispozici bezplatný plugin implementující Python program využívající framework TensorFlow. Tento plugin dále řeší komunikaci mezi prostředími skrz síťové sockety. Navzdory možnosti ovládat plugin prostřednictvím blueprintů v Unreal Engine neexistuje dobře zdokumentovaný návod, jak tento plugin zprovoznit.

Koncept síťových socketů je však pro navázání komunikace mezi Python programem a enginem klíčový. Plugin TCPSocketConnection, který je k dispozici v Marketplace, umožňuje připojení k jednotlivým socketům, odesílání a přijímání zpráv. Plugin obsahuje také vlastní blueprinty a jeho nastavení je poměrně snadné. Plugin však funguje pouze jako klient, takže server pro tuto komunikaci musí být implementován na straně programu Python. Znázornění této komunikace je v obrázku 6.

Pro účely testování byl vytvořen server, který naslouchá zprávám a vypisuje je do konzole. Tento server bude součástí finální verze programu, ve kterém bude přijímat data a vracet jejich výsledky. Pro usnadnění práce se síťovými sockety byl použit balíček socket. Zde je ořezaný pseudokód pro lepší pochopení:

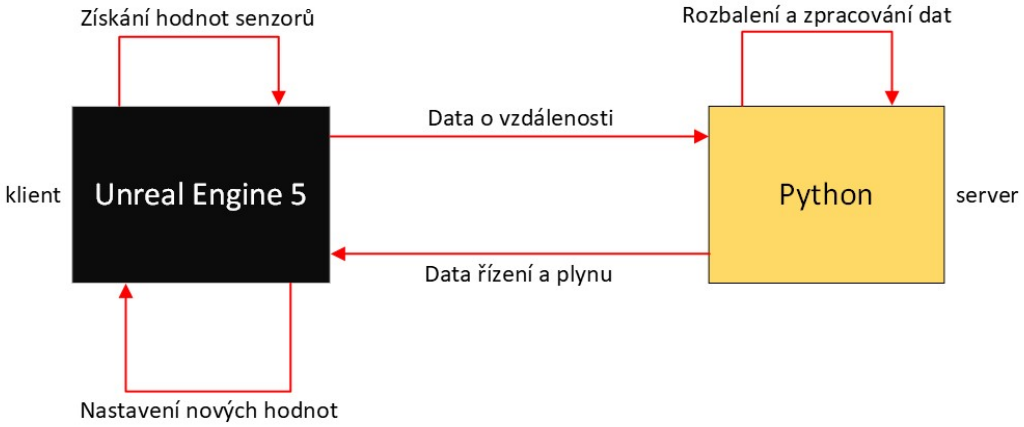
```
import socket
function server(host='127.0.0.1', port=65432)
    with socket as s:
    s.listen(host,port)
    while True:
        conn, addr = s.accept()
        with conn:
            while True:
            data = conn.recv
            print(data)
```

7.1.1 Testování posílání zpráv

Pro použití výše uvedeného pluginu bylo nutné vytvořit novou třídu implementující tento plugin.
V blueprint třídě se nastavil hlavní uzel *Connect* s parametry IP adresy a portu. Dále byly vytvořeny tři nové události pro připojení k serveru, odpojení od serveru nebo příchod zprávy, ke kterým lze připojit další funkce.

Dalším krokem bylo vložení této třídy do světa. To umožní její volání v jiných třídách. V třídě auta byly připraveny senzory a získávání vzdálenosti z původního návrhu. Tento sběr dat je nastaven tak, aby probíhal každých 0,25 sekundy. Pro odesílání zpráv ve strukturovaném formátu bylo nutné z jednotlivých hodnot sestavit formát CSV a teprve poté odeslat řetězec znaků na server. Pro tento problém byla vytvořena funkce *ToCSVFormat*, která přijímá jednotlivé hodnoty typu float a vrací řetězec znaků.

Závěrem lze říci, že každou čtvrtinu sekundy jsou data získána ze snímačů, převedena do formátu CSV a odeslána na server.



Obrázek 6: Proces komunikace[9]

7.2 Tvorba tréninkových dat

Díky této implementaci bylo možné začít vytvářet trénovací data pro neuronovou síť. Proces trénování zahrnuje odesílání dat ze senzorů a dat nastavení úrovně zatáčení a stlačení plynového pedálu. Tato data naučí síť, aby se sama rozhodovala, co nejpřesněji, při jakých vzdálenostech od zdí je potřeba nastavit jakou hodnotu.

Pro začátek bylo zapotřebí duplikovat třídu auta, a to z důvodu jiného druhu posílání dat. Duplikovaná třída byla rozšířena o dodatečné získávání informací ze zmiňovaného nastavení řízení a plynu. Kromě toho bylo potřeba upravit funkci pro vytváření formátu CSV, aby zahrnovala tyto dvě hodnoty.

Server musel být rovněž upraven. Byl rozšířen tak, aby obsahoval funkci, v níž se každá přijatá zpráva vloží jako řádek do souboru CSV. V tomto souboru jsou uložena všechna shromážděná data pro trénování neuronové sítě. Program pro toto trénování se jmenuje *Training*.

[9]Zdroj: vlastní

Trénování

Pro vytváření dat byl použit gamepad, který obsahuje možnost analogového vstupu. Běžná klávesnice by totiž při trénování nastavovala hodnoty řízení a plynu pouze v maximálních hodnotách. Sice by neuronová síť s těmito daty pracovala, ale mnohem lepší pro ni budou data z analogového vstupu, která lépe znázorňují skutečné řízení. Šablona závodní hry již naštěstí podporuje herní zařízení, takže nebylo nutné implementovat další funkce pro podporu gamepadu.

Při trénování bylo nutné použít různorodou škálu dat. Patří sem pomalá a rychlá jízda, pobyt ve středu trati nebo v blízkosti stěn a samozřejmě různé zatáčky. Potenciálním řešením je také trénink od jiných jedinců, kteří hrají hru jinak. Je také nutné vyvarovat se chybných záznamům, jako jsou kolize se stěnami nebo stání na místě.

Pro neuronovou síť byla vytvořena sada přibližně 7500 záznamů. To je pro základní testování dostatečné, ale pro pokročilejší rozhodování umělé inteligence je třeba tento soubor výrazně rozšířit.

V průběhu trénování byly zjištěny nedostatky v senzorech auta. Prvním z nich byl větší mrtvý úhel mezi senzorem 0 a senzory -45, 45. Pro zlepšení a zpřesnění tréninku byly přidány další dva senzory pod úhly 10 a -10 stupňů. Všechny funkce pro přenos dat musely být odpovídajícím způsobem upraveny.

Druhý problém se objevil v případě prudké akcelerace, při které se auto zdvihne a senzor 0 stupňů přesahuje stěnu trati. Tento problém byl vyřešen přidáním neviditelných zdí, se zapnutou kolizí do modelu trati.

7.3 Implementace neuronové sítě

S vytvořeným souborem s tréninkovými daty bylo možné začít vyvíjet samotný program s neuronovou sítí. Aby nebylo nutné trénovat neuronovou síť při každém spuštění programu, byly vytvořeny dva programy.

Jako framework pro neuronové sítě byl zvolen *PyTorch*, protože se snadno instaluje v prostředí *PyCharm* od společnosti *JetBrains*. Zároveň je *PyTorch* snazší pro implementaci jednodušších neuronových sítí než jiné frameworky. Komunitní verze *PyCharm* lze stáhnout zdarma[10]. Prostředí obsahuje i správce přídavných modulů. Zde stačilo pouze zadat jméno balíčku PyTorch a stisknutím tlačítka se framework nainstaloval a byl připraven k použití.

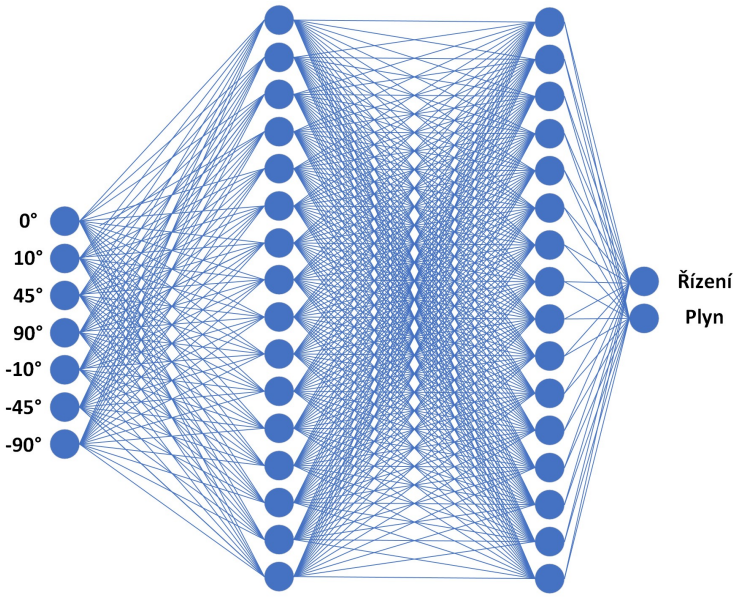
7.3.1 AiDriver

Hlavní program obsahuje server pro komunikaci prostřednictvím socketů, neuronovou síť a zpracování CSV dat.

Navržená neuronová síť (Obrázek 7) má podobu 7 vstupů, dvou skrytých vrstev po 16 neuronech a 2 výstupů. Jako aktivační funkce je zvolena funkce *ReLu*, která vrací výsledek, pokud je kladný, jinak vrací nulu. Jedná se o jednu z nejběžnějších funkcí v neuronových sítích [37].

[10] `https://www.jetbrains.com/pycharm/download/`

Na startu programu se načte uložený vytrénovaný model, který je produktem programu *ModelTrainer*. Poté se spustí server a začne naslouchat. Pokud dorazí zpráva, pokusí se rozdělit CSV formát na pole hodnot. Tyto hodnoty pak pošle do neuronové sítě. Její výsledný výstup zabalí opět do CSV formátu a pošle připojenému klientovi na serveru. Tento proces pokračuje, dokud není program ukončen.



Obrázek 7: Neuronová síť[11]

7.3.2 ModelTrainer

Tento program je určen k trénování neuronové sítě. Jeho výstupem je vytvořený a natrénovaný model neuronové sítě uložený ve speciálním souboru. Tento soubor se pak načte při spuštění hlavního programu *AiDriver*. Tímto způsobem se zrychlí samotné načítání programu a přidá možnost ukládání různých natrénovaných modelů.

Program obsahuje návrh neuronové sítě, který musí odpovídat programu *AiDriver*. Následně program zpracuje přiložený CSV soubor s tréninkovými daty, pomocí kterých vycvičí model neuronové sítě. Po zadání dat se neuronová síť začíná učit. Do konzole se vypisuje výstup, který říká, kolik projití celého souboru dat již program udělal. Toto projití se označuje jako *Epoch*. Zároveň s tím je vypočtena hodnota *Loss*. Ta zase označuje výsledek ztrátové funkce MSE. Nakonec program uloží daný model do speciálního souboru s příponou *pt*.

7.4 Zpětná komunikace

Po otestování výstupních dat neuronové sítě do konzole bylo potřeba tyto výstupy poslat zpět do Unreal Enginu. S tím pomohl opět plugin TCPSocketConnection. Ve třídě implementující tento plugin byla již vytvořena událost příchozí zprávy. Díky vývojové funkci *Print string*, která ve hře vytiskne zprávu na obrazovku, je možné testovat, zda engine zprávy přijímá.

[11] Zdroj: vlastní

Po otestování, zda engine opravdu přijímá zprávy, bylo potřeba jej převést na *float* hodnoty a nastavit je třídě s autem.

Nastavování hodnot

Ve třídě TCPConnection byly vytvořeny proměnné *Steering* a *Throttle*. V případě přijaté zprávy se zpráva rozdělí na jednotlivé hodnoty a přetypují se na *float* hodnotu. Poté se tyto *float* hodnoty nastaví do zmiňovaných proměnných.

Tyto proměnné bylo potřeba předat do třídy *AiCar*. V této třídě byla pro čistotu kódu vytvořena událost. Tato událost si zavolá třídu *TCPConnection* obsaženou ve světě a jejím prostřednictvím si zavolá vytvořené proměnné. Jejich hodnoty jsou pak nastaveny jako řízení a plyn. Tato událost je volána každý *event tick*.

8 Rozšíření hry

Aby program splňoval základní charakteristiky hry, bylo potřeba vylepšit mapu, přidat další prvky závodní hry a vytvořit grafické uživatelské rozhraní pro spouštění.

8.1 Tvorba hlavního menu

Hlavní nabídka tvořené hry řeší problém, kdy je potřeba před startem hry zvolit jaký režim bude hráč hrát. K dispozici jsou tři režimy. Prvním je trénovací režim, ten se uchová v systému v případě budoucího rozšíření dat pro neuronovou síť. Druhý režim slouží k pozorování samostatné umělé inteligence. Tento režim slouží primárně k testování a souhrnu výsledků. Naposledy byl přidán režim pro klasické hraní hry hráčem.

Pro vytvoření grafického rozhraní stačilo pouze vytvořit nový level a speciální widget blueprint, který pracuje s 2D grafikou. Po nastavení levelu, ve kterém se při startu zavolá tento blueprint, přichází samotný návrh rozhraní. I v tomto ohledu Unreal Engine poskytuje jednoduchou implementaci. Díky jeho grafickému návrháři je celý proces *drag and drop*. Po přidání tlačítek k nim lze přiřadit různé události. V tomto případě stačí událost *OnClick*, ve kterém po prokliku systém otevře již známé blueprinty a poté lze nadále přidávat funkce.

Hlavní menu, které lze vidět na obrázku 8, obsahuje čtyři tlačítka, z kterých tři jsou pro volbu režimů a poslední pro ukončení hry. Režimová tlačítka otevřou level s tratí a nastaví, která třída má být použita jako hráč.

Obrázek 8: Hlavní menu hry[12]

Prozatímní implementace otevření a zvolení třídy je nastavena jako tři různé mapy, každá s vlastním herním režimem, ve kterém je nastavena třída, za kterou se má hrát.

[12]Zdroj: vlastní

8.2 Přidání logiky hry

Po vzoru jiných závodních her byla přidána časomíra při závodění. Tím vznikl cíl hry, kterým je dosažení co nejlepšího času na trati. Tato funkce byla přidána pro režim hráče a umělé inteligence. Díky tomu je možné porovnávat umělou inteligenci s hráčem.

Prvním krokem bylo vytvoření objektu, který uchovává čas a také zpracovává kolize. Jedná se o startovní čáru. Byla vytvořena pomocí dvou komponent. Běžný 3D objekt, který slouží především jako vizuální reprezentace čáry a kolizní box, jehož výška je vyšší než zobrazovaný objekt. Prvotní návrh neobsahoval kolizní box a kolize se řešila prostřednictvím startovní čáry. V důsledku toho vznikl problém špatného počítání kol, protože každý průjezd zvedl počet kol dvakrát. To bylo způsobeno nízkou výškou kolize, která registrovala každou nápravu vozu jako samostatný objekt.

Koliznímu boxu byla nastavena událost, která spustí časovač, když detekuje kolizi s hráčem. Hodnota času je uložena v datovém typu *float*, takže je třeba ji převést do textového formátu. K tomu byla vytvořena *macro* funkce, která je volána ve třídě startovní čáry, a výsledný čas je uložen do nové proměnné. Byl vytvořen také druhý časovač, který slouží k ukládání času pro jednotlivá kola. Včetně časovače je k dispozici také událost, která tyto časy odesílá do *VehiclePlayerController*.

Právě tato třída, která má v základu zpracovávání vstupů od uživatele, funguje jako prostředník pro přeposílání časů. Pro zobrazování těchto časů byl vytvořen další widget blueprint. Po vložení textových polí stačilo pouze je propojit se třídou *VehiclePlayerController*, aby si brala text z proměnných.

Jakmile hráč překročí startovní čáru, spustí se časovač závodu a kola. Když hráč dokončí kolo, čas kola se uloží, zobrazí a časovač se vynuluje. Pokud se jedná o poslední kolo a hráč překročí cílovou čáru, celkový čas závodu se zastaví.

9 Souhrn výsledků

První pokus o implementaci umělé inteligence se strojovým učením přímo v prostředí Unreal Engine narazil na několik problémů. S ohledem na to se přešlo k druhému přístupu, který zahrnoval použití frameworku PyTorch v nově vyvinutém programu. Tento přístup však také představoval novou výzvu, a to navázání komunikace mezi Unreal Enginem a programem v jazyce Python. To bylo úspěšně implementováno díky TCP socketům. Díky vyřešené komunikaci se mohlo přejít na vytvoření neuronové sítě a generování trénovacích dat.

V průběhu vytváření tréninkových dat byl vytvořen soubor se zhruba 7500 záznamy. Tento soubor stačil jako základní balíček pro další kroky implementování a zároveň pro testování funkčnosti umělé inteligence. Po vytvoření dat následovalo samotné programování dvou aplikací, které mají na starost umělou inteligenci. První aplikace trénuje modely neuronové sítě a ukládá je do speciálního souboru. Druhý program pouze tento model načte a poté spustí server pro komunikaci s Unreal Enginem, ve kterém přijatá data zpracovává a posílá zpět.

Po vyladění a ověření funkčnosti komunikace bylo možné začít testovat jízdu umělé inteligence. První pokusy byly velmi neúspěšné, veškerý pohyb auta se spíš jevil jako náhodný. To se razantně změnilo úpravou nastavení trénování modelů, ve kterém se mnohonásobně zvýšil počet iterací celé sady dat. Následně již při první jízdě auto bylo schopné reagovat na zatáčky a upravovat svou rychlost podle vzdálenosti od nich. Některé zatáčky umělá inteligence hladce projela vysokou rychlostí, zatímco v jiných případech zpomalila a opatrně manévrovala.

Přesto je při pozorování umělé inteligence patrné, že si na trati stále není vždy jistá. Během testování se občas vyskytly případy, kdy vůz narazil do zdi nebo v horších případech uvízl a nepokračoval v jízdě. Tyto problémy by však mělo vyřešit rozšíření tréninkových dat a další vylepšení celého procesu. Následně by umělá inteligence měla být schopna bez problémů projíždět podobnou trať a to díky rychlé komunikaci mezi prostředími.

9.1 Možná rozšíření

Tento druh implementace nabízí spoustu dalších cest vylepšení a rozšíření. Lze říci, že tento projekt lze využít jako šablona pro pokročilejší vylepšování videohry a umělé inteligence.

Pokud jde o umělou inteligenci, je možné vylepšit neuronovou síť začleněním dalších informací o stylu jízdy. S velkým množstvím dat by se auto řízené neuronovou sítí stalo silným konkurentem člověka. Zlepšit by se mohlo i samotné trénování neuronové sítě. Mohly by být využity genetické algoritmy, podobné těm, které byly použity v původní implementaci.

Počítačová hra byla primárně vytvořena za účelem zkoumání implementace neuronových sítí. Pokud by bylo záměrem vydat tento projekt jako počítačovou hru, bylo by jistě nutné značné rozšíření ve všech aspektech obsahu. Složitější by bylo také zabalit hru spolu s programem v jazyce Python, protože by bylo třeba vymyslet metodu, jak spustit program na pozadí při spuštění hry.

10 Závěr

V práci bylo představeno využití herního enginu Unreal Engine při tvorbě počítačových her. I přes to, že hlavním cílem nebylo vytvořit plnohodnotnou hru, herní engine, včetně jeho šablon, byl plně kompetentní i pro testování možné implementace strojového učení. Zároveň Unreal Engine je velmi přívětivý pro začátečníky, a to díky skriptování skrz blueprinty. Veškeré funkcionality ohledně počítačové hry a komunikace s neuronovou sítí byly implementovány právě skrz toto grafické rozhraní. To usnadnilo celkovou programovací práci.

V další fázi byly představeny dva přístupy využití neuronových sítí v herním enginu. První zahrnoval implementaci strojového učení přímo v Unreal Engine. Toto řešení bylo funkční, ale přinášelo vysokou náročnost na hardware a učení se ukázalo jako ne zcela ideální. V druhém případě byla neuronová síť implementována pomocí frameworku PyTorch v programu v jazyce Python. Problematická byla komunikace mezi těmito dvěma prostředími. To bylo vyřešeno pomocí komunikace přes TCP sockety, ve kterých program Python sloužil jako server a Unreal Engine, s TCP komunikačním pluginem, jako klient. Toto řešení bylo poměrně úspěšné, komunikace probíhala rychle a spolehlivě. Pro toto řešení byl vytvořený soubor tréninkových dat, která potvrzují funkčnost komunikace a učení neuronové sítě.

Tato práce dále otevírá i další možnosti zkoumání této problematiky. Zajímavým rozšířením by byla tvorba přídavného pluginu do Unreal Enginu, který by implementoval Python program obsahující PyTorch. Po průzkumu knihoven pluginů bylo nalezeno pár implementací, ty jsou často ale pro starší verze Unreal Engine, nebo jejich použití vyžaduje programování v C++. Větším oponentem by byl již zmiňovaný plugin využívající TensorFlow. Ten bohužel nese spoustu neznámých o jeho implementaci a neexistuje žádná obsáhlá dokumentace, jak tento plugin spustit a využít. Pokud by se možný plugin s PyTorch dobře zdokumentoval, mohl by se stát silným nástrojem a ulehčením pro začínající vývojáře.

I myšlenka tvorby autonomního vozidla se dá rozšířit. Možné cesty jsou skrz implementování strojového učení do robotického auta a vytvoření trati ve fyzickém světě. S tím by bylo nutné i přejít na jiný styl sběru dat. Bylo by nutné snímat trať kamerou a obraz z ní, pomocí počítačového vidění, zpracovávat pro informace o poloze auta na dráze.

Odvětví automatizování řízení, umělé inteligence a počítačových her je zajímavé. Jejich propojení v rámci práce rozšířily znalosti a naskytly možnosti, jak dále tyto obory zkoumat.

Literatura

[1] HORBAN, Oleksandr; MARTYCH, Ruslana; MALETSKA, Mark. Phenomenon of Videogame Culture in Modern Society. *Studia Warmińskie*. 2020, roč. 56, s. 123–135. Dostupné z DOI: 10.31648/sw.4314.

[2] *Beyond Traditional Play: The Transformative Impact of Generative AI in Gaming* [online]. 2024. [cit. 2024-04-15]. Dostupné z: https://medium.com/@DigitalQuill.ai/comparative-analysis-traditional-ai-generative-ai-in-the-game-industry-39144dbe0b67.

[3] *Video Game History - Timeline & Facts* [online]. 2022. [cit. 2023-07-22]. Dostupné z: https://www.history.com/topics/inventions/history-of-video-games.

[4] MONTELLI, Chrissy. What does 'NPC' mean? Understanding non-player characters, an important aspect of any video game. *Business Insider* [online]. 2021 [cit. 2023-08-25]. Dostupné z: https://www.businessinsider.com/guides/tech/npc-meaning.

[5] QAFFAS, Alaa A. An Operational Study of Video Games' Genres. *International Journal of Interactive Mobile Technologies (iJIM)*. 2020, roč. 14, č. 15. Dostupné z DOI: 10.3991/ijim.v14i15.16691.

[6] ROUSE, Margaret. *Sandbox Game* [online]. 2024. [cit. 2024-04-03]. Dostupné z: https://www.techopedia.com/definition/3952/sandbox-gaming.

[7] JAVADI, Kia. *Most Played Games 2023: A Look Back At The Top Gaming Obsessions - GadgetMates* [online]. 2024. [cit. 2024-04-04]. Dostupné z: https://gadgetmates.com/most-played-games-2023. Section: Gaming.

[8] A History of Arcade Games. *Betson Enterprises* [online]. 2021 [cit. 2023-08-25]. Dostupné z: https://www.betson.com/the-history-of-arcade-games/. Section: Blog.

[9] WEATHERBED, Jess. PC gaming is booming right now, but where does it go from here? *TechRadar* [online]. 2021 [cit. 2023-08-25]. Dostupné z: https://www.techradar.com/news/pc-gaming-is-booming-right-now-but-where-does-it-go-from-here.

[10] MONTES DE OCA, Bernardo. The rise, fall, and uncertain future of eSports [online]. 2023 [cit. 2023-08-25]. Dostupné z: https://slidebean.com/story/the-rise-fall-and-uncertain-future-of-esports.

[11] CURRY, David. *Most Popular Mobile Games (2024)* [online]. 2024. [cit. 2024-04-04]. Dostupné z: https://www.businessofapps.com/data/most-popular-mobile-games/.

[12] The Famous 9 Video Game Console Generations - Plarium. *plarium.com* [online]. 2022 [cit. 2023-08-10]. Dostupné z: https://plarium.com/en/blog/console-generations/.

[13] INFORMER, Digit. The Evolution of the Gaming Steering Wheel - DigitInformer.Com [online]. 2023 [cit. 2023-08-25]. Dostupné z: https://www.digitinformer.com/the-evolution-of-the-gaming-steering-wheel/. Section: General Info.

[14] SHEA, Ryan; LIU, Jiangchuan; NGAI, Edith C.-H.; CUI, Yong. Cloud gaming: architecture and performance. *IEEE Network*. 2013, roč. 27, č. 4. Dostupné z DOI: 10.1109/MNET.2013.6574660.

[15] STEFYN, Nadia. How Video Games Are Made. *cgspectrum* [online]. 2022 [cit. 2024-04-13]. Dostupné z: https://www.cgspectrum.com/blog/game-development-process.

[16] LEWIS, Michael; JACOBSON, Jeffrey. Game engines. *Communications of the ACM*. 2002, roč. 45, č. 1, s. 27. Dostupné také z: https://www.cse.unr.edu/~sushil/class/gas/papers/GameAIp27-lewis.pdf.

[17] *Unity* [online]. 2024. [cit. 2024-04-12]. Dostupné z: https://unity.com/.

[18] *CryEngine* [online]. 2024. [cit. 2024-04-12]. Dostupné z: https://www.cryengine.com/.

[19] *Unreal Engine* [online]. 2024. [cit. 2024-04-12]. Dostupné z: https://www.unrealengine.com/en-US/home.

[20] *Introduction to Blueprints* [online]. 2024. [cit. 2024-04-05]. Dostupné z: https://dev.epicgames.com/documentation/en-us/unreal-engine/introduction-to-blueprints-visual-scripting-in-unreal-engine.

[21] JOHNS, Robert. *Unity vs Unreal: Which Game Engine Should You Choose?* [online]. 2024. [cit. 2024-04-05]. Dostupné z: https://hackr.io/blog/unity-vs-unreal-engine.

[22] SHEIKH, Haroon; PRINS, Corien; SCHRIJVERS, Erik. Artificial Intelligence: Definition and Background. In: *Mission AI: The New System Technology*. Cham: Springer International Publishing, 2023, s. 15–16. ISBN 978-3-031-21448-6. Dostupné z DOI: 10.1007/978-3-031-21448-6_2.

[23] *A definition of AI*. European Commission, 2018. Dostupné také z: https://ec.europa.eu/futurium/en/system/files/ged/ai_hleg_definition_of_ai_18_december_1.pdf.

[24] GLOVER, Ellen. *Strong AI vs. Weak AI: What's the Difference? | Built In* [online]. 2024. [cit. 2024-04-05]. Dostupné z: https://builtin.com/artificial-intelligence/strong-ai-weak-ai.

[25] OPPY, Graham; DOWE, David. The turing test. 2003. Dostupné také z: https://plato.stanford.edu/entries/turing-test/#ChiRoo.

[26] DONGARE, AD; KHARDE, RR; KACHARE, Amit D et al. Introduction to artificial neural network. *International Journal of Engineering and Innovative Technology (IJEIT)*. 2012, roč. 2, č. 1, s. 189–194. Dostupné také z: https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=04d0b6952a4f0c7203577afc9476c2fcab2cba06.

[27] SHARKAWY, Abdel-Nasser. Principle of Neural Network and Its Main Types: Review. *Journal of Advances in Applied & Computational Mathematics*. 2020, roč. 7, s. 8–19. Dostupné z DOI: 10.15377/2409-5761.2020.07.2.

[28] YAMASHITA, Rikiya; NISHIO, Mizuho; DO, Richard Kinh Gian; TOGASHI, Kaori. Convolutional neural networks: an overview and application in radiology. *Insights into Imaging*. 2018, roč. 9, č. 4, s. 611–629. ISSN 1869-4101. Dostupné z DOI: 10.1007/s13244-018-0639-9.

[29] GROSSBERG, Stephen. Recurrent neural networks. *Scholarpedia*. 2013, roč. 8, č. 2, s. 1888. Dostupné také z: http://scholarpedia.org/article/Recurrent_neural_network.

[30] *What is a Neural Network?* [online]. 2024. [cit. 2024-04-05]. Dostupné z: https://www.ibm.com/topics/neural-networks.

[31] *Genetic Algorithms* [online]. 2017. [cit. 2024-04-19]. Dostupné z: https://www.geeksforgeeks.org/genetic-algorithms/. Section: DSA.

[32] KATOCH, Sourabh; CHAUHAN, Sumit Singh; KUMAR, Vijay. A review on genetic algorithm: past, present, and future. *Multimedia Tools and Applications*. 2021, roč. 80, č. 5, s. 8091–8126. ISSN 1573-7721. Dostupné z DOI: 10.1007/s11042-020-10139-6.

[33] DARBINYAN, Rem. Council Post: How Artificial Intelligence Can Empower The Future Of The Gaming Industry. *Forbes* [online]. 2022 [cit. 2023-08-25]. Dostupné z: https://www.forbes.com/sites/forbestechcouncil/2022/07/13/how-artificial-intelligence-can-empower-the-future-of-the-gaming-industry/. Section: Innovation.

[34] *TensorFlow* [online]. 2024. [cit. 2024-04-19]. Dostupné z: https://www.tensorflow.org/.

[35] *PyTorch* [online]. 2024. [cit. 2024-04-19]. Dostupné z: https://pytorch.org/.

[36] *Keras: Deep Learning for humans* [online]. 2024. [cit. 2024-04-19]. Dostupné z: https://keras.io/.

[37] BROWNLEE, Jason. *A Gentle Introduction to the Rectified Linear Unit (ReLU)* [online]. 2019. [cit. 2024-03-18]. Dostupné z: https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/.

Univerzita Hradec Králové
Fakulta informatiky a managementu

Zadání bakalářské práce

Autor:	**Jiří Dostál**
Studium:	I2100194
Studijní program:	B1802 Aplikovaná informatika
Studijní obor:	Aplikovaná informatika
Název bakalářské práce:	**Tvorba počítačové hry a její řešení pomocí umělé inteligence**
Název bakalářské práce AJ:	Creating of a video game and its solution using artificial intelligence

Cíl, metody, literatura, předpoklady:

Cílem práce je navrhnout a následně vytvořit umělou inteligenci a vytrénovat ji tak, aby byla schopna vyřešit jednoduchou počítačovou hru. Pro tyto účely bude hra vytvořena v herním enginu Unreal Engine 5.

Tereotická část:
Definice a prvky počítačové hry
Umělá inteligence(true AI)
Umělá inteligence ve hrách(fake AI)
Herní enginy

Praktická část:
Tvorba počítačové hry
Vytvoření Umělé inteligence, která bude danou hru řešit

Zadávající pracoviště:	Katedra informačních technologií, Fakulta informatiky a managementu
Vedoucí práce:	Ing. Milan Kořínek

Datum zadání závěrečné práce: 15.10.2021